

AD-A069 029

TRW DEFENSE AND SPACE SYSTEMS GROUP REDONDO BEACH CALIF
LAMAS SYSTEM MANUAL. (U)

FEB 78

F/G 9/2

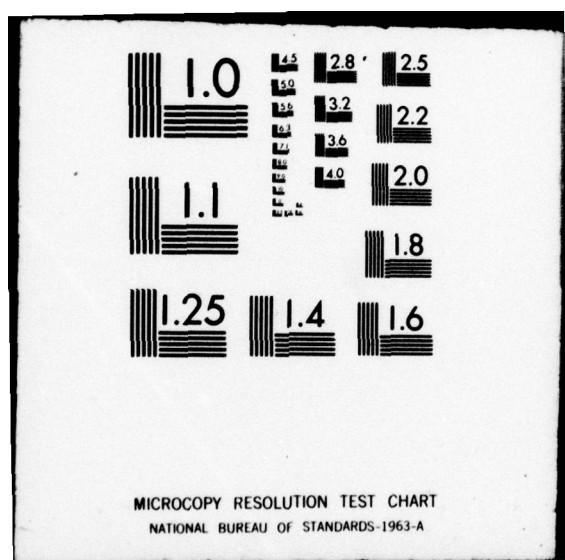
DAAG39-77-C-0112

NL

UNCLASSIFIED

| OF
AD
A069 029





DDC FILE COPY

LEVEL II

AD A069029

ADVANCED STUDIES
SANTA BARBARA, CALIFORNIA 93106

LAMAS SYSTEM MANUAL

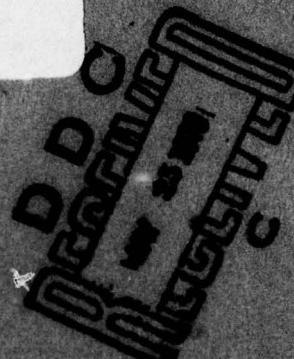
00000000000000000000000000000000

FEBRUARY 1978

CONTRACT NO. DDC-ES-77-C-0112

This document has been approved
for public release and sale in
whole or in part.

John T. Austin
Chief of Staff



1
vw

10 00 09 147

DISCLAIMER NOTICE

**THIS DOCUMENT IS BEST QUALITY
PRACTICABLE. THE COPY FURNISHED
TO DDC CONTAINED A SIGNIFICANT
NUMBER OF PAGES WHICH DO NOT
REPRODUCE LEGIBLY.**

TRW.

AND SPACE SYSTEMS GROUP

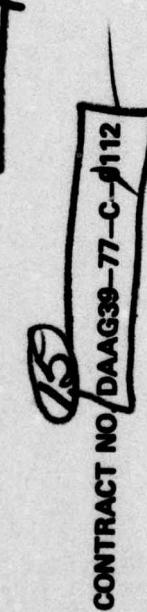
ADVANCED STUDIES

ONE SPACE PARK • REDONDO BEACH • CALIFORNIA 90278

LAMAS SYSTEM MANUAL



CDRL ITEM A008



CONTRACT NO DAAG39-77-C-1112

Prepared for:
TRADOC Combined-Arms Test Activity
U.S.A. Intelligence Center and School

This document has been approved
for public release and sale; its
distribution is unlimited.

409631

79 04 09 14 7

LAMAS SYSTEM MANUAL

This document describes the LAMAS software package from both a programmer and user standpoint. The text is written in Program Design Language (PDL) format and represents the final version of the working document used throughout program development. All LAMAS software and manual procedures are described, with the exception of the display software developed using company funds. A proprietary version of this document which describes both deliverable and company-funded software is maintained on file in the TRW Advanced Studies office.

2

↑

ACCESSION FOR	White Section	<input type="checkbox"/>
	Buff Section	<input type="checkbox"/>
NTIS	Announced	<input checked="" type="checkbox"/>
DOC	Classification	<input checked="" type="checkbox"/>
8- <i>See</i>		
DISTRIBUTION AND AVAILABILITY		
CROS		
SPECIAL		
239		

R

TRW SYSTEMS GROUP
THE SPACE PARK
REEDLEY BLANCH, CALIFORNIA 90278
(213) 535-4321

* * * * *
LOCATION AND MOVEMENT ANALYSIS SYSTEM

(LAMAS)

* * * * *
SOFTWARE TO COMPUTE GROUND FORCE MOVEMENT
* * * * *
CONSIDERING MOVEMENT DICTRINE, ROAD CONDITIONS,
* * * * *
AND RISK FACTORS

* * * * *
21 FEB 78

* * * * *
PDL 03.06

TABLE OF CONTENTS

PAGE 1.001

GENERAL DESCRIPTION	2
OVERVIEW	3
INITIALIZATION	4
GROUND MOVEMENT CALCULATIONS	5
USER FORMAT	6
SYSTEM ARCHITECTURE	7
DATA BASE DESCRIPTIONS	8
NODES AND LINKS	9
NODE VECTOR FORMAT	10
NODE VECTOR DESCRIPTION	11
VERBAN	12
CROSS-COUNTRY AND CONCEALMENT CODE DESCRIPTIONS	13
UNITS	14
UNIT-VECTOR FORMAT	15
UNIT-VECTOR DESCRIPTION	16
SOLUTIONS	17
SOLUTION-VECTOR FORMAT	18
SOLUTION-VECTOR DESCRIPTION	19
FACTS	20
ROUTE-VECTOR FORMAT	21
ROUTE-VECTOR DESCRIPTION	22
MAP DIRECTORY	23
CROSS-COUNTRY DIRECTORY	24
PREPARATION FOR USE	25
PREPARATION	26
CREATE NODE-VECTOR AND MAP-DIRECTORY FILES ON DISK	27
CREATE NODE CARD	28
NODE CARD FORMAT	29
CREATE LINK CARD	30
LINK CARD FORMAT	31
CREATE MAP-DIRECTORY ENTRY FOR THIS MAP	32
CREATE CROSS-COUNTRY, CONCEALMENT, AND DIRECTORY FILES ON DISK	33
MAPS FILE	34

TRI, INC.
21 FEB 78

LOCATION AND MOVEMENT ANALYSIS SYSTEM
TABLE OF CONTENTS

TABLE OF CONTENTS

	PAGE
GENERAL DESCRIPTION	1.001
OVERVIEW	2
INITIALIZATION	3
GROUND MOVEMENT CALCULATIONS	4
USER FORMAT	5
SYSTEM ARCHITECTURE	6
DATA BASE DESCRIPTIONS	7
NODES AND LINKS	8
NODE VECTOR FORMAT	9
NODE VECTOR DESCRIPTION	10
TERRAIN	11
CROSS-COUNTRY AND CONCEALMENT CODE DESCRIPTIONS	12
UNITS	13
UNIT+VECTOR FORMAT	14
INIT+VECTOR DESCRIPTION	15
SOLUTIONS	16
SOLUTION+VECTOR FORMAT	17
SOLUTION+VECTOR DESCRIPTION	18
ROUTES	19
ROUTE+VECTOR FORMAT	20
ROUTE+VECTOR FORMAT DESCRIPTION	21
MAP DIRECTORY	22
CROSS-COUNTRY DIRECTORY	23
PREFARATION FOR USE	24
PREPARATION	25
CREATE NODE+VECTOR AND MAP+DIRECTORY FILES ON DISK	26
CREATE NODE CARD	27
NODE CARD FORMAT	28
CREATE LINK CARD	29
LINK CARD FORMAT	30
CREATE MAP+DIRECTORY ENTRY FOR THIS MAP	31
CREATE CROSS-COUNTRY, CONCEALMENT, AND DIRECTORY FILES ON DISK	32
LAMAS FILE	33
	34

LAMAS	35
IMPLEMENT ALGORITHMS AND FUNCTIONS USING THE NODE+VECTOR DATA BASE	36
PREPARE FOR ROAD NETWORK PATH CALCULATIONS	37
INITIALIZE MAP+NUMBERS AND SCREEN EXTREMES	38
ESTABLISH DIRECTORY IN MEMORY AND READ IN NODE+VECTORS	39
READ THIS MAP'S NODES INTO MAIN MEMORY	40
CHANGE ADJACENT NODE'S MAP+NUMBER TO INDEX	41
INITIALIZE UNIT+VECTORS	42
ESTABLISH A FORWARD EDGE OF BATTLE AREA	43
FIND NODE+VECTOR NEAREST TO UTM COORDINATES	44
PATH DETERMINATION AND DISPLAY	45
PERFORM INTERDICTIVE OPERATIONS	46
PRINT MAP+NUMBER, NODE+VECTOR NEAREST GIVEN COORDINATES	47
CHANGE CONTENTS OF A NODE+VECTOR	48
CHANGE CHARACTERISTIC	49
DELETE A NODE FROM THE NODE+VECTOR ARRAY	50
PRINT CONTENTS OF A NODE+VECTOR	51
CHANGE PROPERTIES OF A UNIT	52
CALCULATE PATH	53
PURGE ALL ROUTES	54
REINITIALIZE "WORKING LIST" NODE+VECTOR ENTRIES	55
CALCULATE AND ASSIGN TIME AND MARCH VALUES	56
COMPUTE NEXT NODE TO LABEL	57
EXPLANATION OF PATH DETERMINATION ALGORITHM	58
COMPUTE SOLUTION VECTORS AND ROUTE+VECTOR	59
CALCULATE SECOND BEST PATH	60
FIND THE ORDER OF PRIORITIES FOR BEST NETWORK TRAVEL	61
CALCULATE BEST MODE AT WHICH TO INTERDICT	62
PRESENT RESULTS	63
PRINT TABLE OF ROUTE NUMBERS WITH ASSOCIATED UNIT NAMES	64
PRINT PATH STATISTICS	65
IMPLEMENT ALGORITHMS AND FUNCTIONS USING THE CROSS-COUNTRY DATA BASE	66
PERFORM TERRAIN INITIALIZATION	67
MAP+NUMBER TO BE USED	68
ESTABLISH UNIT+VECTORS	69
CHANGE ENTERED COORDINATES TO AN INDEX INTO THE DATA BASE	70
PERFORM TERRAIN PATH CALCULATIONS	71
CALCULATE CROSS-COUNTRY PATH	72
OBTAIN UNIT NAMES TO BE CONSIDERED	73
OBTAIN THE PATH PARAMETERS	74
CALCULATE THE CROSS-COUNTRY PATH	75

TRI- INC. LOCATION AND MOVEMENT ANALYSIS SYSTEM
21 FEB 78 TABLE OF CONTENTS

	PAGE
CALCULATE THE ADJACENT NODE'S TIME VALUES	76
CONSTRUCT SOLUTION-VECTORS AND ROUTE-VECTOR	77
DISPLAY RESULTS	78
DATA INDEX	79
FLOW SEGMENT INDEX	80
SEGMENT REFERENCE TREES	81
	1.003

TRU, INC.
21 FEB 78

LOCATION AND MOVEMENT ANALYSIS SYSTEM

PAGE 2

* GENERAL DESCRIPTION *

TRW, INC.
21 FEB 78

LOCATION AND MOVEMENT ANALYSIS SYSTEM
GENERAL DESCRIPTION

OVERVIEW

PAGE 3

1 THE PURPOSE OF THIS SOFTWARE PACKAGE IS TO COMPUTE GROUND
2 FORCE MOVEMENT USING OPTIMAL PATH DETERMINATION AND ROUTE CONFLICT
3 RESOLUTION ALGORITHMS. TO INSURE FLEXIBILITY, USER INTERACTION IS
4 SUPPORTED DURING MOST PHASES OF THE SOFTWARE'S OPERATION.
5 THREE MAJOR AREAS ARE INVOLVED IN THE PROCESS: INITIALIZATION
6 GROUND FORCE MOVEMENT CALCULATIONS, AND DISPLAY.

10
11
12
13
14
15

INITIALIZATION

0 1 INITIALIZATION IS THE PROCESS OF PREPARING DATA BASES FOR
2 GENERAL USE. THERE ARE THREE BASIC TYPES OF DATA BASES: PERMA-
3 NENT, TEMPORARY - USER DEFINED, AND TEMPORARY - PROGRAM DEFINED.
4 PERMANENT DATA BASES ARE CREATED BEFORE THE USER EVER
5 INVOKES THE SYSTEM, AND ARE GENERALLY UNALTERABLE BY THE USER.
6 INCLUDED ARE SUCH FILES AS NODES, ROADS, CITIES, CROSS-COUNTRY MOVE-
7 MENT, CONCEALMENT, MAP+DIRECTORY, ROAD+DIRECTORY, AND CROSS-COUNTRY
8 DIRECTORY.
9 A USER-DEFINED TEMPORARY DATA BASE IS A FILE OVER WHICH THE
10 USER HAS CONSIDERABLE CONTROL, BUT WHICH EXISTS ONLY FOR A PAR-
11 TICULAR SESSION ON THE SYSTEM. ONCE THE USER SIGNS OFF, THE DATA
12 BASE IS LIST. THE UNITS DATA BASE IS THE PRIMARY EXAMPLE OF THIS
13 TYPE IN LAMAS. HERE THE USER DEFINES THE CONTENTS, AND MAY ALTER
14 THE CONTENTS AT ANY TIME, BUT THE FILE DOES NOT CARRY OVER FROM
15 ONE SESSION TO ANOTHER.
16 PROGRAM-DEFINED TEMPORARY DATA BASES ARE FILES OVER WHICH
17 THE USER HAS NO CONTROL WHATSOEVER. THE FILES MAY ENABLE THE USER
18 TO DISPLAY SOME RESULTS, BUT NO ALTERATIONS MAY BE MADE. EX-
19 AMPLES OF THIS TYPE OF DATA BASE ARE SOLUTION+VECTORS, ROUTE+
20 VECTORS, MAP+DIRECTORY IN MEMORY, AND ROAD+DIRECTORY IN MEMORY.
21 THE EFFECT OF INITIALIZATION VARIES FROM FILE TO FILE.
22 FOR SOME, IT IS JUST SETTING ALL ELEMENTS TO ZERO, AS IN THE CASE
23 OF THE SOLUTION+VECTORS AND ROUTE+VECTORS. FOR OTHERS, IT IS THE
24 ESTABLISHING OF VALUES, AS IN ALL OF THE PERMANENT FILES, UNITS DATA
25 BASE, AND THE DIRECTORIES.
26 THE MANNER IN WHICH THE VALUES ARE SET ALSO VARIES FROM
27 FILE TO FILE. ALL PERMANENT FILES ARE CREATED BY TRANSFERRING
28 INFORMATION FROM DATA CARDS TO DISK STORAGE. THE UNITS FILE IS
29 ESTABLISHED BY DIRECT USER INPUT, WHILE THE NODE+VECTOR FILE
30 IN MEMORY AND THE DIRECTORIES IN MEMORY ARE CREATED BY THE PRO-
31 GROM, BUT ONLY AFTER THE USER HAS INVOKED PARTICULAR ROUTINES.

0

GROUND MOVEMENT CALCULATIONS

1 THERE ARE TWO BASIC TYPES OF GROUND MOVEMENT CALCULATIONS WE WILL MAKE IN THIS PACKAGE. ONE CONSIDERS THE
2 ROAD NETWORK FOR THE AREA OF INTEREST AS REPRESENTED IN THE
3 NODE+VECTOR DATA BASE, WHILE THE OTHER USES CROSS-COUNTRY
4 MOVEMENT INFORMATION AS SUPPLIED BY THE ARMY ON ITS OCN AND
5 CONCEALMENT MAPS FOR THE AREA OF INTEREST.
6 THE MAJOR DIFFERENCES BETWEEN THESE TWO APPROACHES
7 ARE IN THEIR SCOPE OF APPLICATION, AND THE DATA BASES WHICH
8 SUPPLY THE INFORMATION. USING THE ROAD NETWORK APPROACH, ONE
9 MAY CONSIDER UP TO TEN 1:50000 MAPS FOR ANY COMPUTATION, WHILE
10 THE CROSS-COUNTRY MOVEMENT MUST BE LIMITED TO AT MOST ONE
11 1:50000 MAP. THIS IS DUE TO THE MANNER IN WHICH THE TWO TYPES
12 ARE MODELED.
13
14 BECAUSE OF THE DIFFERENCE IN INFORMATION BEING HELD IN
15 THE DATA BASE REGARDING THE TWO TYPES OF MOVEMENT, THIS CALLS
16 FOR TWO VERY DIFFERENT DATA BASE APPROACHES. THESE WILL BE
17 EXPLAINED LATER ON IN THIS DOCUMENT.
18 ON THE OTHER HAND, THE ALGORITHM USED TO CALCULATE THE
19 PATHS FOR EACH TYPE OF MOVEMENT IS VIRTUALLY IDENTICAL. THE
20 ONLY DIFFERENCE LIES IN THE INTERPRETATION OF THE DATA BASES;
21 THE ACTUAL MECHANICS OF THE PATH CALCULATIONS ARE THE SAME.
22 THE ALGORITHM FINDS THE "BEST" PATH FOR A UNIT TO TRAVEL.
23 "BEST" IS A USER-DEFINED PARAMETER MADE UP OF TWO COMPONENTS,
24 RISK AND TIME. RISK IS A VALUE ARRIVED AT BY CONSIDERING
25 ROAD CONDITIONS, TERRAIN FACTORS, WHETHER OR NOT A BRIDGE IS
26 PRESENT, AND WHETHER OR NOT A CITY IS PRESENT. TIME IS THE
27 AMOUNT OF TIME IT TAKES A PARTICULAR UNIT TO TRAVEL FROM ONE
28 NODE TO ANOTHER.
29 OTHER TYPES OF PATH CALCULATIONS MAY BE PERFORMED AS
30 WELL. THE USER MAY FIND THE SECOND BEST PATH TO TRY TO ATTACK
31 THE SENSOR TASKING PROBLEM, HE MAY FIND THE BEST NODE AT WHICH
32 TO PERFORM INTERDICTION FOR DISRUPTION PURPOSES, OR HE MAY FIND
33 THE BEST LONGEST PATH (A METHOD OF CALCULATING HOW QUICKLY A FORCE
34 MAY BUILD UP IN A PARTICULAR AREA.

HSSE & FONCTION

1 THE FORMAT FOR USER INTERACTION WILL BE OF A SIMPLE NATURE.
2 WHEN STARTING, A BASIC MENU OF AVAILABLE FUNCTIONS WILL BE LISTED ON
3 A DISPLAY SCREEN. THE USER WILL DECIDE WHICH ONE HE WISHES TO PURSUE,
4 AND ENTER ITS CODE AT A TERMINAL. THIS MAY CAUSE DIRECT ACTION, OR
5 CAUSE ANOTHER MENU TO BE DISPLAYED. THROUGHOUT THE OPERATION, WHETHER
6 OR NOT A MENU APPEARS, THE USER WILL BE PROMPTED WITH INSTRUCTIONS
7 REGARDING HOW TO PROCEED. IN THIS WAY, THE USER HAS "HANDS-ON" CONTROL
8 OF ANY AND ALL ACTIVITIES INVOLVED IN THE GROUND FORCE MOVEMENT COMPUTATIONS.

SYSTEM ARCHITECTURE

0 1 THIS SOFTWARE HAS BEEN DESIGNED TO OPERATE ON A PDP 11/45 COMPUTER
0 2 HAVING 96K WORDS OF MEMORY, WITH ANY SINGLE TASK ABLE TO ACCESS AT MOST
0 3 32K WORDS. THE OPERATING SYSTEM, RSX-11M, SUPPORTS A USER DEFINED
0 4 MULTITASKING CAPABILITY, WITH THE TOTAL AMOUNT OF MEMORY USAGE TO NOT
0 5 EXCEED 64K WORDS.

0 6 FOUR DIFFERENT PERIPHERALS ARE NEEDED TO EFFECTIVELY USE
0 7 THIS SOFTWARE PACKAGE. ONE IS A TERMINAL FOR COMMUNICATION WITH THE
0 8 PROGRAM. THIS MAY BE A DECWRITER, A TEKTRONIX CRT, OR A VTOS TERMINAL.
0 9 ALL PERMANENT DATA BASES ARE STORED IN AN RF04 REVIVING HEAD DISK UNIT, AS
0 10 WELL AS ALL OF THE SOURCE AND OBJECT CODE, AND TASK IMAGE. A GOULD 4800
0 11 HIGH-SPEED LINE PRINTER IS USED FOR PATH STATISTICS OUTPUT, AND A CONTROL
0 12 8500 IMAGE PROCESSING SYSTEM IS USED FOR THE VISUAL DISPLAYS.

0 13 DUE TO MEMORY CONSIDERATIONS, SUCH AS THE LARGE AMOUNT OF SPACE
0 14 NEEDED FOR DATA BASES, IT HAS BEEN DECIDED TO USE MULTIPLE TASKING TO
0 15 IMPLEMENT THIS PROGRAM. ONE TASK PERFORMS ALL OF THE I/O FUNCTIONS, THE
0 16 OBTAINING OF USER INPUT AND THE PRINTING OF MESSAGES, WHILE ANOTHER TUES
0 17 ALL OF THE COMPUTATION. RSX-11M IS STRUCTURED SO THAT THIS MAY BE DONE
0 18 WITH A MINIMUM OF DELAY TIME USED TO SWITCH FROM ONE TASK TO THE OTHER.
0 19 MESSAGES MAY BE SENT BETWEEN THE TWO, AND THIS IS HOW OUTPUT DATA IS
0 20 TRANSFERRED. THERE IS ALSO A SEPARATE GRAPHICS PACKAGE WHICH IS ITSELF
0 21 ANOTHER TASK. SO THERE ARE THREE TASKS RUNNING CONCURRENTLY IN THIS SYS-
0 22 TER.

0 23 THE TOTAL MEMORY OF THESE THREE TASKS MUST NOT EXCEED 64K WORDS,
0 24 OR SEVERE DEGRADATION OF RESPONSE TIME IS INCURRED. THIS NECESSITATES
0 25 THE USE OF OVERLAYS IN ORDER TO SAVE MEMORY SPACE.
0 26 WE HAVE ARRIVED AT THESE ESTIMATES FOR STORAGE REQUIREMENTS:

DESCRIPTION	CONTENTS	MEMORY USED
0 29		121
0 30		122
0 31	IN-CORE NODE-ARRAY (ABOUT 10 MAPS)	123
0 32		124
0 33		125
0 34	SOLUTION-VECTORS	126
0 35	40 ROUTES WITH 25 NODES PER ROUTE.	127
0 36		128
		129

TRI, INC.
21 FEB 78

LOCATION AND INVENTENT ANALYSIS SYSTEM
GENERAL DESCRIPTION

PAGE 7.001

	OTHER	DIRECTORIES, COUNTERS, ETC.	5,000 WORDS	#
# 37				130
# 38				# 131
# 39			22,800 WORDS	# 132
# 40				# 133
# 41				# 134
# 42				135
# 43				# 136
# 44				# 137
# 45				# 138
# 46				139
# 47				# 140
				#

THIS APPARENTLY LEAVING US WITH 10,000 WORDS FOR ACTUAL EXECUTABLE CODE PER DISPLAY. HOWEVER, THE GRAPHICS PACKAGE USES 21,300 WORDS, AND THE I/O ROUTINES USE APPROXIMATELY 17,000 WORDS, SO ONLY ABOUT 4,000 WORDS ARE TRULY AVAILABLE FOR CODING IN THE MAIN ROUTINE.

TIMINGS: TO BE DETERMINED

TRW, INC.
21 FEB '8

LOCATION AND MOVEMENT ANALYSIS SYSTEM

PAGE: 8

* DATA BASE DESCRIPTIONS *
* *****

NODES AND LINKS

1 CONSISTING OF ONLY NODE-VECTORS. THIS DATA BASE IS THE LARGEST, AND # 143
2 PROBABLY THE MOST IMPORTANT, OF ALL THE DATA BASES. THE INFORMATION # 144
3 CONTAINED WITHIN ENABLES THE PATH ALGORITHM TO ACCURATELY CALCULATE THE # 145
4 BEST PATH FOR A GIVEN UNIT TO TRAVEL. # 146
5 A NODE-VECTOR IS A 32-MORD STORAGE AREA WHICH CONTAINS DATA PER- # 147
6 TRAINING TO A NODE AND THE LINKS WHICH CONNECT IT TO OTHER NODES. A NODE # 148
7 IS A POINT OF INTEREST, USUALLY, BUT NOT NECESSARILY, BEING AT THE JUNC- # 149
8 TION OF TWO OR MORE ROADS. THE LINK BETWEEN TWO NODES IS MOST OFTEN A # 150
9 REPRESENTATION OF THE ROADWAY JOINING THE TWO, BUT IT CAN ALSO BE A REP- # 151
10 PRESENTATION OF SIMPLY THE GENERAL PATH (CROSS-COUNTRY OR OTHERWISE) # 152
11 WHICH CONNECTS THEM. TWO NODES ARE SAID TO BE ADJACENT TO EACH OTHER # 153
12 IF THERE EXISTS A SINGLE LINK BETWEEN THEM. A PATH IS THEN A SERIES # 154
13 OF LINKS SUCH THAT THE LINKS, WHEN TAKEN AS A WHOLE, ARE CONNECTED. THAT # 155
14 IS, STARTING AT NODE 1, A LINK CONNECTS WITH NODE 2 ADJACENT TO NODE 1, # 156
15 WHICH CONNECTS WITH NODE 3 ADJACENT TO NODE 2, ETC., UNTIL THE DESTI- # 157
16 NATION NODE IS REACHED. ALL NODE-VECTOR DATA HAS BEEN OBTAINED FROM # 158
17 STANDARD L SERIES 1:50000 MAPS. # 159
18 A GROUND FORCE UNIT'S MOVEMENT THROUGH A ROAD NETWORK IS CALCULATED # 160
19 BY MODELING ITS TRAVEL FROM ONE NODE TO ANOTHER ALONG A LINK. A LINK # 161
20 IS A CHARACTERIZATION OF A PATH, END AS SUCH, PASSES ALONG ASSOCIATED # 162
21 PIECES OF INFORMATION, SUCH AS: DISTANCE, NUMBER OF LANES, ROAD TYPE, # 163
22 OFF-ROAD TRAFFICABILITY, AND TERRAIN, BRIDGE, AND CITY CODES. # 164
23 DISTANCE REFERS TO THE ACTUAL DISTANCE IN KILOMETERS ONE WOULD # 165
24 HAVE TO TRAVEL TO MOVE FROM ONE NODE TO ANOTHER ALONG THE TRUE CONNEC- # 166
25 TION. NUMBER OF LANES MEANS THE NUMBER OF SINGLE-FILE ROAD LANES # 167
26 WHICH ARE EQUIVALENT TO THE LINKING ROAD. ROAD TYPE IS A CODE REFLECT- # 168
27ING ANY OF FOUR POSSIBILITIES: AUTOSTRADE, MAIN ROAD, SEC- # 169
28 ONARY ROAD, OR FAIR WEATHER ROAD ONLY. OFF-ROAD TRAFFICABILITY IS AND- # 170
29 OTHER CODE, THIS TIME INDICATING OPEN TRAFFICABLE AREAS NEAR THE ROAD # 171
30 WHICH WILL ALLOW REST OR ALTERNATE ROUTES. TERRAIN MEASURES LAND FEA- # 172
31 TURES AND INTERVISIBILITY. THE CHOICES ARE: FLAT, HILLY, MOUNTAINOUS, OR # 173
32 ANY OF THESE THREE PLUS HIGH TERRAIN IMMEDIATELY TO THE WEST. THE # 174
33 BRIDGE CODE INDICATES THE PRESENCE, OR LACK OF, A BRIDGE, AND IF IT EX- # 175
34 ISTS, ITS SIZE, EITHER SMALL, MEDIUM, OR LARGE. FINALLY, CITY CODE # 176
35 FUNCTIONS IN A MANNER IDENTICAL TO THE BRIDGE CODE. RISK IS TAKEN TO # 177
36 BE A LINEAR COMBINATION OF OFF-ROAD TRAFFICABILITY, TERRAIN, BRIDGE, # 178

AND CITY CODES.

38 THE CROSS-COUNTRY AND CONCEALMENT PATH ALGORITHM DOES NOT USE
39 THE NODES AND LINES DATA BASE.

40 THE ELEMENTS WHICH CHARACTERIZE A NODE ARE SIGNIFICANTLY DIFFERENT
41 THAN THOSE CHARACTERIZING A LINK. ALL NODAL INFORMATION IS LOCATION
42 ORIENTED, THAT IS, IT DESCRIBES EXACTLY WHERE THE NODE IS ON THE MAP.
43 THIS INFORMATION INCLUDES LATITUDE, LONGITUDE, THE MAP NUMBER FOR THE
44 PARTICULAR MAP ON WHICH THE NODE RESIDES, AND THE NODE NUMBER (AN ARB-
45 ITRARY FIGURE ESTABLISHED DURING THE MANUAL CREATION OF THE DATA BASE.)
46 A NODE+VECTOR THEN, IS COMPRISED OF THE ABOVE-MENTIONED NODAL AND
47 LINK INFORMATION (A MAXIMUM OF FOUR ADJACENT NODES MAY EXIST), PLUS
48 SEVERAL OTHER VALUES, WHOSE SIGNIFICANCE WILL BECOME CLEAR LATER ON IN
49 THIS DOCUMENT (AMONG THESE ARE NORTH MEASURE, PREDECESSOR NODE, CUM-
50 MULATIVE TIME, CUMULATIVE NORTH MEASURE, TIME MEASURE, SOLUTION VECTOR
51 POINTER, LANE USED, RISK MEASURE, AND PARK TIME. THESE ARE ALL USED BY
52 THE PATH ALGORITHM.)

53 NODE+VECTORS MAKE UP A PERMANENT DATA BASE WHICH RESIDES ON DISK.

54 ONE MAY THINK OF THIS STORAGE AS BEING DIVIDED INTO BINS: SUCH THAT ONE
55 BIN CONTAINS THE NODE+VECTORS FOR ONE MAP. A DIRECTORY IS ALSO STORED
56 ON DISK, AND IT CONTAINS IDENTIFICATION FOR EACH BIN, THAT IS, THE AP-
57 PROPRIATE MAP NUMBER. ACCESS TO ANY MAP'S NODE+VECTORS IS QUITE SIMPLE;
58 ONE FINDS THE MAP NUMBER IN THE DIRECTORY, THE DIRECTORY ENTRY POINTS
59 TO THE BIN ON DISK WHICH CONTAINS THE NODE+VECTORS, WHICH MAY THEN
60 BE READ INTO MAIN MEMORY.

61 ONCE IN MAIN MEMORY, NODE+VECTORS ARE ALTERABLE BY THE USER.
62 FUNCTIONS EXIST WHICH ALLOW THE USER TO CHANGE CERTAIN VALUES: SUCH AS
63 LINK DISTANCE, ROAD TYPE, OR NUMBER OF LANES. ANY CHANGES MADE ARE NOT
64 PERMANENT: SINCE THE "CORRECTED" VERSION OF THE DATA BASE IS NEVER
65 COPIED INTO THE PERMANENT DISK FILE.

66 WHILE IN MEMORY, THE NODE+VECTORS ARE THOUGHT OF AS OCCUPYING
67 A NODE+VECTOR ARRAY. MAIN MEMORY IS LINEARLY ORGANIZED, SO IT IS NAT-
68URAL TO THINK OF NODE+VECTORS COMPRISING A LIST INDEXABLE BY INTEGERS.
69 THUS, IF TWO MAP'S NODE+VECTORS ARE IN MEMORY, WITH THE FIRST MAP HAVING
70 30 NODE+VECTORS, THE FIRST NODE+VECTOR OF THE SECOND MAP IS THE 31 ST
71 NODE+VECTOR IN THE ARRAY, ETC.

72 *****
73 *****
74 *****
75 *****
76 *****
77 *****
78 *****
79 *****
80 *****
81 *****
82 *****
83 *****
84 *****
85 *****
86 *****
87 *****
88 *****
89 *****
90 *****
91 *****
92 *****
93 *****
94 *****
95 *****
96 *****
97 *****
98 *****
99 *****
100 *****
101 *****
102 *****
103 *****
104 *****
105 *****
106 *****
107 *****
108 *****
109 *****
110 *****
111 *****
112 *****
113 *****
114 *****
115 *****
116 *****
117 *****
118 *****
119 *****
120 *****
121 *****
122 *****
123 *****
124 *****
125 *****
126 *****
127 *****
128 *****
129 *****
130 *****
131 *****
132 *****
133 *****
134 *****
135 *****
136 *****
137 *****
138 *****
139 *****
140 *****
141 *****
142 *****
143 *****
144 *****
145 *****
146 *****
147 *****
148 *****
149 *****
150 *****
151 *****
152 *****
153 *****
154 *****
155 *****
156 *****
157 *****
158 *****
159 *****
160 *****
161 *****
162 *****
163 *****
164 *****
165 *****
166 *****
167 *****
168 *****
169 *****
170 *****
171 *****
172 *****
173 *****
174 *****
175 *****
176 *****
177 *****
178 *****
179 *****

NODE VECTOR FORMAT

NOTE THAT EACH WORD IS SIXTEEN BITS,
AND IT IS OFTEN POSSIBLE TO CONSIDER TWO
PIECES OF INFORMATION FOR ONE WORD, WITH
EACH PIECE OCCUPYING ONE BYTE. IF THERE
IS NO SECOND INFORMATION PIECE IN A WORD,
THE DATA IS RIGHT-ADJUSTED WITHIN THE WORD.

WORD # BYTES 1 2
1 NODE MAP+NUMBER
2 NODE # WITHIN MAP
3 NORTH MEASURE
4 PRECESSOR NODE NUMBER
5 CUMULATIVE TIME
6 CUMULATIVE NORTH MEASURE
7 TIME MEASURE
8 SOLUTION VECTOR POINTER
9 LATITUDE
10 LONGITUDE
11 NUMBER OF ADJACENT NODES
12 ADJACENT MAP #1
13 NODE+NUMBER WITHIN MAP
14 # OF LANES
15 OFF-ROAD TRAFFICABILITY
16 BRIDGE
17 ADJACENT MAP # 2
18 NODE # WITHIN MAP
19 # OF LANES
20 OFF-ROAD TRAFFICABILITY
21 BRIDGE
22 ADJACENT MAP #3
23 NODE # WITHIN MAP
24 # OF LANES
25 OFF-ROAD TRAFFICABILITY
26 BRIDGE
27 ADJACENT MAP #4
28 NODE # WITHIN MAP
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250

TRW, INC.
21 FEB 78

LEADER:INN AND MINE-LEVEL DATA ANALYSIS SYSTEM

PAGE 10.001

4	37	29	4	41
4	38	30	4	42
4	39	31	4	43
4	40	32	4	44

251 252 253 254

卷之三

NODE VECTOR DESCRIPTION

WORD #	DESCRIPTION AND USE	RANGE
1	NUMBER OF 1:50000 MAP FROM WHICH DATA IS TAKEN.	1-32767
2	NUMBER OF NODE WITHIN THE MAP. THESE NUMBERS ARE SELECTED WHEN THE NODES ARE MANUALLY DEFINED.	1-80
3	FLAG TO INDICATE WHETHER OR NOT THIS NODE HAS BEEN LABELED BY THE PATH ALGORITHM.	0-1
4	NORTH OF THE LINK BETWEEN THIS NODE AND ITS PREDECESSOR, AS DETERMINED BY THE PATH ALGORITHM.	0-100
5	NODE+VECTOR INDEX OF THE PREDECESSOR NODE+VECTOR, SELECTED BY THE OPTIMAL MOVE ALGORITHM.	1-400
6	TOTAL NORTH MEASURE OF THE PATH FROM INITIAL NODE UP TO AND INCLUDING THIS LINK'S WERTH MEASURE (WORD 3).	0-32767
7	THE ACTUAL TIME FOR A UNIT TO PROGRESS FROM PREDECESSOR TO THIS NODE.	0-15639
8	INDEX TO ENTRY IN THE SOLUTION-VECTOR ARRAY. THIS POINTER ENABLES THE NODE+VECTOR TO BE ASSOCIATED WITH THE OPTIMAL ROUTE DATA. ONLY NODES THAT ARE PART OF AN OPTIMAL SOLUTION WILL HAVE THIS ENTRY. IF THE NODE IS PART OF SEVERAL OPTIMAL ROUTES, THIS WORD WILL POINT TO THE	0-5000
9		286
10		287
11		288
12		289
13		290
14		291

TRW, INC.
21 FEB 78

LOCATION AND MOVEMENT ANALYSIS SYSTEM DATA BASE DESCRIPTIUNS

PAGE 11,001

MIST RECENTLY CREATED BY VECIAG.

# 38	9	NEST RECENTLY CALCULATED SOLUTION-VECTOR.	
# 39	9	NODE LATITUDE IN KILOMETERS FROM EQUATOR. THIS DATA IS ONLY USED FOR DISPLAY AND IS THUS + DR - 1 KM RESOLUTION. ROUTE ALGORITHMS USE LINK DISTANCES, NOT THESE LATITUDE, LONGITUDE COOR- DINATES.	0-32767
# 40	10	NODE LONGITUDE IN KILOMETERS FROM GREENWICH. THIS DATA IS ONLY USED FOR DISPLAY AND IS THUS + DR - 1 KM RESOLUTION.	0-32767
# 41	10	NUMBER OF ADJACENT NODES, MAXIMUM IS FOUR, MINIMUM IS ONE.	1-4
# 42	11,1	LANE NUMBER USED DURING PATH CALCULATION. THIS IS NECESSARY IN ORDER TO EFFECTIVELY USE MULTIPLE LANED ROADS.	1-4
# 43	11,2	THE MAP-NUMBER OF THE FIRST ADJACENT NODE. THIS ENTRY WILL BE RESET AFTER DATA IS READ FROM DISK TO CURE TO BECOME THE NODE-VECTOR ARRAY INDEX OF THE FIRST ADJACENT NODE.	1-32767
# 44	12	NUMBER OF NODE WITHIN THE MAP OF THE FIRST ADJACENT NODE.	1-80
# 45	13,1	LINK DISTANCE IN TENTHS OF A KM BETWEEN THE NODE AND THE FIRST ADJACENT NODE.	0-255
# 46	13,2	NUMBER OF SINGLE FILE ROAD LANES BETWEEN THE NODE AND THE FIRST ADJACENT NODE.	1-4
# 47	14,1	ROAD TYPE CODE. 1=AUTOBAHN OR AUTOSTRASE, 2=MAIN ROAD, 3=SECONDARY ROAD OR ROAD, 4= FAIR WEATHER ROAD ONLY.	1-4
# 48	14,2	GROUND COVER CODE TO INDICATE OPEN TRAFFIC-	1-5
# 49	15,1		

75 ABLE AREAS NEAR ROAD THAT ALLOWS REST OR ALTER-
76 RATE PATHS.
/> 1=URBAN, 2=CULTIVATED, 3=MUNICIPAL,
78 4=SWAMP,
79 5=ROAD NOT ADJACENT TO MAIN LINK.

80 15,2 TERRAIN. CODE TO INDICATE TERRAIN AND COARSE-
81 INTERVISIBILITY MEASURE. 1=FLAT, 2-HILLY,
82 3=MOUNTAINOUS, 10 HIGH TERRAIN IMMEDIATELY TO THE
83 WEST, AND ANY OF THE FIRST THREE PLUS 10.

84 16,1 BRIDGE. 0=NONE, 1=SMALL, 2=MEDIUM, 3=LARGE.
85 # 86 0-3
87 16,2 CITY. 0=NONE, 1=SMALL, 2=MEDIUM, 3=LARGE
88 0-3
89 17-21 IDENTICAL TO WORDS 12-16 BUT APPLICABLE TO THE
90 SECOND ADJACENT NODE, IF PRESENT.
91 # 92 22-26 IDENTICAL TO WORDS 12-16 BUT APPLICABLE TO THE
93 THIRD ADJACENT NODE, IF PRESENT.
94 # 95 27-31 IDENTICAL TO WORDS 12-16 BUT APPLICABLE TO THE
96 FOURTH ADJACENT NODE, IF PRESENT.

97 # 98 32,1 RISK MEASURE FOR THE LINK BETWEEN THIS NODE AND
99 ITS PREDECESSOR. RISK IS TAKEN TO BE THE SUM OF
60 GROUND COVER, TERRAIN, BRIDGE, AND CITY COSES.
61 # 62 32,2 USED BY THE PATH ALGORITHM TO REMEMBER HOW MANY
02 MINUTES A UNIT PARKED AT THIS NODE.
#

***** TOO MANY LINES IN SEGMENT

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

#

TERRAIN

1 THIS DATA BASE SUPPORTS THE TERRAIN MODEL FOR PATH CALCULATIONS.
2 THERE ARE ACTUALLY THREE DATA BASES INVOLVED, CROSS-COUNTRY MOVEMENT,
3 SUMMER CONCEALMENT, AND WINTER CONCEALMENT.
4 EACH DATA BASE SUPPLIES DATA FOR A SPECIFIC TYPE OF PATH CALCULA-
5 TION. CROSS-COUNTRY YIELDS DATA FOR FASTEST MOVEMENT, AND THE TWO CON-
6 CEALMENT DATA BASES GIVE INFORMATION ABOUT THE RISK OF A PATH.
7 THESE DATA BASES WERE CREATED BY PLACING A 93 X 88 SQUARE GRID OVER
8 THE DATA MAPS SUPPLIED BY THE ARMY. EACH SQUARE WAS THEN ASSIGNED A
9 VALUE CORRESPONDING TO THE AREA WHICH THE SQUARE COVERED. THESE VAL-
10 UES WERE TRANSFERRED TO COMPUTER CARDS WHICH WERE THEN USED TO CREATE
11 THREE DATA FILES ON DISK, "CCM.DAT", "SCCM.DAT", AND "WCMM.DAT".
12 BECAUSE OF THE REGULARITY OF THE SQUARES, EACH SQUARE MAY BE RE-
13 GARDED AS A NODE WITH EIGHT NEIGHBORS (UNLESS IT LIES ON AN EDGE). THUS,
14 EACH MAP IS REPRESENTED BY 8184 NODES. FOR ANY NODE, NO MORE INFORMA-
15 TIONS IS NEEDED, SINCE ALL LINK ATTRIBUTES ARE KNOWN.
16 THERE IS, HOWEVER, PATH INFORMATION WHICH MUST BE STORED DURING
17 A CALCULATION, BUT THERE SIMPLY ISN'T ENOUGH MEMORY TO ALLOCATE 5 WORDS
18 FOR EACH OF THE 8184 NODES. THUS, WHEN USING THE TERRAIN MODEL FOR
19 PATH CALCULATIONS, THERE IS AN ADDITIONAL DATA BASE CALLED THE "WORKING
20 LIST". AS A NEW NODE IS INCLUDED IN THE CALCULATION, A NEW ENTRY TO
21 THE WORKING LIST IS MADE. SINCE NOT ALL NODES WILL BE INCLUDED, THE
22 WORKING LIST'S SIZE IS CONSIDERABLY SMALLER THAN 8184 * 5 WORDS, AND
23 FITS IN MAIN MEMORY WITH EASE.
24 ANOTHER SPACE-SAVING DEVICE IS EMPLOYED. THE CODE FOR ANY SQUARE
25 IS NEVER GREATER THAN 15, SO DATA FOR FOUR NODES MAY BE PLACED IN ONE
26 WORD OF MEMORY, THUS 8184 CODES USE ONLY 2046 WORDS.

360 # 361
362 # 363
364 # 365
366 # 367
368 # 369
370 # 371
372 # 373
374 # 375
376 # 377
378 # 379
380 # 381
382 # 383
384 # 385

TRI, INC.
21 FEB 78

LOCATION AND MOVEMENT ANALYSIS SYSTEM
DATA BASE DESCRIPTIONS

CROSS-COUNTRY AND CONCEALMENT CODE DESCRIPTIONS

PAGE 13

#	CODE	ESTIMATED MAX. SPEED OF XM-1 TANK (MPH)	MANEUVERABILITY DESCRIPTOR	#
# 1		>= 25	EXCELLENT	387
# 2		20 - 25	VERY GOOD	388
# 3		10 - 20	GOOD	389
# 4		5 - 10	FIR	390
# 5		< 5	POOR	391
# 6			NE GE	392
# 7				393
# 8				394
# 9				395
# 10				396
# 11				397
# 12				398
# 13				399
# 14				400
# 15				401
# 16				402
# 17				403
# 18				404
# 19				405
# 20				406
# 21	CODE	PERCENT CHANCE OF BEING OBSERVED FROM THE AIR		407
# 22		0 - 25 (BEST CONCEALMENT)		408
# 23	8	25 - 50		409
# 24	9	50 - 75		410
# 25	10	75 - 100 (POOREST CONCEALMENT)		411
# 26	11			412

UNITS

0 # 1 THIS IS A TEMPORARY DATA BASE, THAT IS, A DATA BASE WHICH MAY
0 # 2 CHANGE FROM USER TO USER, AND, IN FACT, FROM ONE PATH CALCULATION TO
0 # 3 ANOTHER, IF DESIRED. BECAUSE OF ITS TEMPORARY NATURE, IT EXISTS ONLY
0 # 4 IN MAIN MEMORY; THERE IS NO UNITS DATA BASE ON DISK.
0 # 5 THE UNIT-VECTOR IS THE SOLE CONSTITUENT WITHIN THIS DATA BASE. IT
0 # 6 CONTAINS INFORMATION PERTAINING TO A UNIT, SUCH AS ITS NAME, STARTING
0 # 7 LOCATION, DESTINATION LOCATION, PRESENT LOCATION, PRIORITY, START TIME,
0 # 8 AND TYPE CODE. EACH LOCATION IS ENTERED BY THE USER AS A MAP+NUMBER,
0 # 9 NODE+NUMBER PAIR, WHERE THE MAP+NUMBER IS THE NUMBER OF THE 1:50000 MAP
0 # 10 ON WHICH THE NODE LIES, AND THE NODE+NUMBER IS THAT NUMBER ARBITRARILY
0 # 11 ASSIGNED WHEN THE NODE+VECTOR DATA BASE WAS CONSTRUCTED. HOWEVER, FOR
0 # 12 PURPOSES OF STORAGE, THESE LOCATIONS ARE REPRESENTED INTERNALLY AS IN-
0 # 13 DEXES INTO THE NODE+VECTOR ARRAY. PRIORITY IS A NUMBER REPRESENTING
0 # 14 A UNIT'S IMPORTANCE WITH RESPECT TO OTHER UNITS. START TIME IS A VALUE
0 # 15 ENTERED BY A PATH ALGORITHM, NOT THE USER, AND REPRESENTS MINUTES OF
0 # 16 TIME. TYPE CODE IS A NUMBER WHICH STANDS FOR THE PARTICULAR TYPE OF
0 # 17 UNIT BEING CONSIDERED, SUCH AS A RIFLE BATTALION, OR A MOTORIZED DIVI-
0 # 18 SION.
0 # 19 THIS DATA BASE MAY BE RECREATED EACH TIME A USER WISHES TO PER-
0 # 20 FORM A PATH CALCULATION, AND MAY CONTAIN ANYWHERE FROM 1 TO 60 SEPARATE
0 # 21 UNITS. IF NO UNITS ARE ESTABLISHED BY THE USER, THE PATH ALGORITHMS
0 # 22 WILL NOT FUNCTION, AND AN ERROR MESSAGE PRINTED.

#####

TRW INC.
21 FEB 78

LOCATION AND MOVEMENT ANALYSIS SYSTEM
DATA BASE DESCRIPTIONS

PAGE: 15

UNIT+VECTOR FORMAT

#	#	WORD #	BYTE 1	BYTE 2	#
#	#	1	FIRST LETTER	SECOND LETTER	437
#	#	2	THIRD LETTER	FOURTH LETTER	438
#	#	3	FIFTH LETTER	SIXTH LETTER	439
#	#	4	SEVENTH LETTER	EIGHTH LETTER	440
#	#	5	STARTING LOCATION	PRESENT LOCATION	441
#	#	6	DESTINATION LOCATION	TYPE CODE	442
#	#	7	PRESENT LOCATION	/	443
#	#	8	PRIORITY	/	444
#	#	9	START TIME	/	445
#	#	10	SPARE	/	446
#	#	11			447
#					

TRW, INC.
21 FEB 78

LOCATION AND MOVEMENT ANALYSIS SYSTEM
DATA BASE DESCRIPTIONS

INTRODUCTION

PAGF 16

1	WORD #	DESCRIPTION AND USE	ANY RELIABLE
2	1-4	THIS IS THE UNIT'S NAME IN CHARACTER FORMAT. THERE MAY BE AT MOST EIGHT CHARACTERS.	
3	5	AN INDEX INTO THE NODE+VECTOR ARRAY. THUS, THIS WORD SAYS THAT IF THE INDEX EQUALS I, THE PROPER NODE IS THE I TH NODE IN THE ARRAY.	1-400
4	6	AN INDEX INTO THE NODE+VECTOR ARRAY, INDICATING THE DESTINATION OF THIS UNIT.	1-400
5	7	AN INDEX INTO THE NODE+VECTOR ARRAY, INDICATING WHERE THE UNIT IS AT THIS MOMENT IN THE CALCULATION.	1-400
6	8,1	PRIORITY OF THIS UNIT. 1 IS THE HIGH-EST, 127 IS THE LOWEST. 0 INDICATES THAT THE PRIORITY HAS NOT BEEN ESTABLISHED.	0-1???
7	8,2	CODE INDICATING GENERAL TYPE OF UNIT.	0-255
8	9	TIME AT WHICH UNIT IS KNOWN TO BE AT EITHER ITS STARTING OR FINISHING NODE. THIS DEPENDS UPON MOVEMENT TYPE.	0-15839
9	10	SPARE	

SOLUTIONS

1 ANOTHER TEMPORARY DATA BASE, THIS IS CREATED BY THE PATH ALGORITHM
2 AS IT RECONSTRUCTS A ROUTE OF TRAVEL. EACH SOLUTION+VECTOR SO CREATED
3 CONTAINS INFORMATION ABOUT A PARTICULAR NODE; ITS NODE+VECTOR INDEX,
4 TIME IN, TIME OUT, LANE NUMBER, AND PARK TIME, AND FURTHER DIAGNOSTIC
5 DATA: A POINTER TO ANOTHER SOLUTION+VECTOR WHICH CONSIDERS THE SAME
6 NODE+VECTOR (IF NONE, THE POINTER IS NULL), AND WHETHER OR NOT THIS SOLU-
7 UTION+VECTOR SHOULD BE USED BY THE PATH DECONFLICTING LOGIC.
8 A NODE+VECTOR INDEX IS A POINTER TO A NODE+VECTOR IN MAIN MEMORY.
9 TIME IN AND TIME OUT INDICATE A TIME PERIOD DURING WHICH THE NODE+VEC-
10 TOR IS BUSY. LANE NUMBER KEEPS TRACK OF WHICH LANE THIS UNIT USED TO
11 TRAVEL THROUGH THIS NODE. THIS ENABLES MULTIPLE LANE ROADS TO BE USED
12 BY MORE THAN ONE UNIT AT A TIME. PARK TIME SHOWS HOW MUCH TIME (IN MIN-
13 UTES) THE UNIT HAD TO WAIT AT THIS NODE'S PREDECESSOR BEFORE TRAVELING
14 TO THIS NODE.
15 THE SOLUTION+VECTOR POINTER ALLOWS THE DECONFLICTING LOGIC TO OPER-
16 ATE. BY FOLLOWING THE POINTERS, IT IS POSSIBLE TO FIND ALL INTERVALS
17 DURING WHICH A PARTICULAR NODE IS BUSY. IN THIS WAY, DECONFLICTING, THE
18 PROCESS OF MAKING SURE THAT NO TWO UNITS USE A NODE DURING OVERLAPPING
19 TIME INTERVALS, MAY BE ACCOMPLISHED. THE USER HAS THE OPTION TO INCLUDE
20 OR EXCLUDE THE DECONFLICTING LOGIC. DATA WITHIN THE SOLUTION+VECTOR
21 REFLECTS THE USER'S CHOICE.
22 THE ROUTE DESCRIBED BY A PATH IS A SERIES OF SOLUTION+VECTORS
23 HAVING A ONE-TO-ONE CORRESPONDENCE WITH THE NODES OF THE PATH. IF A
24 PATH'S NODES WERE NUMBERED 1-N, ITS SOLUTION+VECTORS WOULD BE NUM-
25 BERED M-1-M-N, ASSUMING M SOLUTION+VECTORS ALREADY EXISTED AT THE TIME
26 THE PATH WAS CALCULATED. NODE 1 WOULD CORRESPOND TO SOLUTION+VECTOR
27 M-1, NODE 2 WOULD CORRESPOND TO SOLUTION+VECTOR M-2, ETC.

TRW INC.
21 FEB 78

LOCATION AND MOVEMENT ANALYSIS SYSTEM
DATA BASE DESCRIPTIONS

SOLUTION-VECTOR FORMAT

#	WORD #	CONTENTS
0	1	WORD #
0	2	
0	3	1 NODE NUMBER
0	4	2 TIME IN
0	5	3 TIME OUT
0	6	4 MULTIPLE USE NODE POINTER TO ENTRY IN THIS LIST.
0	7	5 +,- LANE NUMBER; PARK TIME
0		

PAGE 18

SOLUTION+VECTOR DESCRIPTION

#	WORD #	DESCRIPTION AND USE	RANGE
# 1	0 2	INDEX NUMBER OF THE ASSOCIATED NODE+VECTOR.	517
# 3	0 3	SEVERAL ROUTES FOR DIFFERENT UNITS MAY POINT	518
# 4	0 4	TO THE SAME NODE+VECTOR.	519
# 5	0 5		520
# 6	0 6	THIS IS THE SIMULATED TIME THAT THE UNIT ENTERS THE	521
# 7	0 7	NUDE AS DETERMINED DURING CONSTRUCTION OF THE SOLUTION+VECTOR. TIME UNITS ARE MINUTES.	522
# 8	0 8	0-14400 MIN.	523
# 9	0 9		524
# 10	0 10	THIS IS THE SIMULATED TIME THAT THE UNIT LEAVES THE	525
# 11	0 11	NUDE. IT IS COMPUTED AS A COMPANION WITH THE ABOVE	526
# 12	0 12	WORD #2. THE UNITS ARE IDENTICAL.	527
# 13	0 13		528
# 14	0 14	THIS NODE+ARRAY INDEX POINTS TO ANOTHER ENTRY IN	529
# 15	0 15	THIS SOLUTION VECTOR LIST WHICH IS ASSOCIATED WITH	530
# 16	0 16	THE SAME NODE+VECTOR. THIS LINK SHOWS THE MULTIPLE USE	531
# 17	0 17	OF A NUDE BY SEVERAL ROUTES OF DIFFERENT UNITS. CHECKING	532
# 18	0 18	THE TIMES IN AND OUT (WORDS 2 AND 3) OF THE MULTIPLE USERS	533
# 19	0 19	OF THE NUDE WILL INDICATE WHETHER OR NOT A CONFLICT HAS	534
# 20	0 20	_OCCURRED.	535
# 21	0 21		536
# 22	0 22	THIS IS THE LANE NUMBER WHICH THIS UNIT USED TO TRAVEL	537
# 23	0 23	THROUGH THIS NUDE.	538
# 24	0 24	1-4	539
# 25	0 25		540
# 26	0 26	TIME IN MINUTES SPENT PARKING AT THE PREDECESSOR NUDE	541
# 27	0 27	BEFORE TRAVELING TO THIS NUDE.	542
# 28	0 28	A POSITIVE VALUE INDICATES THE VECTOR SHOULD BE CON-	543
# 29	0 29	SIDERED WHEN DE-COMPETING. IF NEGATIVE, IT MUST BE	544
# 30	0 30	SKIPPED.	545
	0		546

ROUTES

1 A PATH HAS BEEN DEFINED TO BE A SERIES OF NODES THROUGH WHICH
2 A UNIT TRAVELS IN ORDER TO GET FROM ONE SPECIFIC NODE TO ANOTHER. A
3 ROUTE+VECTOR CONTAINS INFORMATION REGARDING A PARTICULAR ROUTE. THIS
4 INCLUDES A POINTER TO THE UNIT WHICH IS TRAVELING ALONG THE PATH, A
5 POINTER TO THE FIRST SOLUTION+VECTOR FOR THE ROUTE, TOTAL MARCH TIME,
6 TOTAL RISK MEASURE FOR THE PATH, TOTAL PATH DISTANCE, NUMBER OF SOLU-
7 TION+VECTORS IN THE ROUTE, TYPE OF MOVEMENT (FORWARD OR BACKWARD IN
8 TIME), AND INITIAL TIME.
9 THE UNIT POINTER IS AN INDEX INTO THE UNITS DATA BASE, THE SOL-
10 UTION+VECTOR POINTER IS AN INDEX INTO THE SOLUTIONS DATA BASE, AND INITI-
11 AL TIME IS EITHER THE START TIME OR FINISH TIME, AS DICTATED BY THE
12 MOVEMENT TYPE.
13 A ROUTE+VECTOR IS CREATED IMMEDIATELY AFTER THE SOLUTION+VECTORS
14 ARE GENERATED FOR A PARTICULAR ROUTE. THE ROUTES DATA BASE IS THEN
15 TEMPORARY, AND EXISTS ONLY IN MAIN MEMORY. THE USER HAS NO CONTROL
16 OVER THE DATA PLACED WITHIN A ROUTE+VECTOR, AND CAN ONLY KNOW ITS
17 CONTENTS THROUGH INDIRECT MEANS. ONE ROUTINE LISTS UNIT NAMES AND
18 ASSOCIATED ROUTES. ANOTHER LISTS PATH DIAGNOSTICS, SUCH AS TOTAL
19 TRAVEL TIME, TOTAL DISTANCE, AND TOTAL RISK MEASURE, ALONG WITH OTHER
20 INFORMATION. EACH OF THESE USES THE ROUTES DATA BASE AS A SOURCE.
#####

TRI, INC.
21 FEB 78

LOCATION AND MOVEMENT ANALYSIS SYSTEM
DATA BASE DESCRIPTIONS

PAGE 21

ROUTE VECTOR FORMAT

#	# 1	WORK #	CONTENTS	#
# 2	1		UNIT NUMBER	569
# 3	2		LIST HEAD POINTER	570
# 4	3		TOTAL ROUTE TIME	571
# 5	4		TOTAL ROUTE RISK MEASURE	572
# 6	5		TOTAL ROUTE DISTANCE	573
# 7	6		NUMBER OF SOLUTION VECTORS IN ROUTE	574
# 8	7		MOVEMENT TYPE	575
# 9	8		INITIAL TIME	576
# 10	9			577
# 11	10			578
# 12	11			

TECH. INC.
21 FEB 78

LOCATION AND MOVEMENT ANALYSIS SYSTEM
DATA BASE DESCRIPTIONS

PAGE 22

ROUTE+VECTOR FORMAT DESCRIPTION

#	WORD #	DESCRIPTION AND USE	RANGE
# 2	1	INDEX TO UNIT DATA BASE, SO THAT WE KNOW WHICH UNIT USES THIS ROUTE.	1-60
# 4	2	INDEX TO THE SOLUTION VECTOR LIST, POINTING TO THE FIRST SOLUTION VECTOR WHICH HAS THIS ROUTE NUMBER.	1-5000
# 5	3	TOTAL ELAPSED TIME OF MARCH FROM STARTING NODE TO DESTINATION NODE, INCLUDING RESTS AND STOPS, IN MINUTES.	0-15329 MIN.
# 6	4	TOTAL CUMULATIVE RISK MEASURE OF ROUTE	0-3276.7
# 7	5	TOTAL ROUTE DISTANCE, IN TENTHS OF KMS.	0-3276.7
# 8	6	TOTAL NUMBER OF NODES VISITED	1-400
# 9	7	CODE INDICATING THAT THIS WAS DETERMINED USING A STARTING TIME (<=0) OR AN ARRIVAL TIME (>1).	0 OR 1
# 10	8	TIME, IN MINUTES, WHICH THE USER DESIGNATED AS EITHER A STARTING TIME OR AN ARRIVAL TIME, DICTATED BY MOVEMENT TYPE.	0-15839
# 11	9		602
# 12	10		603
# 13	11		604
# 14	12		605
# 15	13		
# 16	14		
# 17	15		
# 18	16		
# 19	17		
# 20	18		
# 21	19		
# 22	20		
# 23	21		
# 24	22		
# 25	23		
# 26	24		

MAP DIRECTORY

0 1 THIS DATA BASE KEEPS TRACK OF THE MAPS DATA BASE. THERE IS ONE
0 2 ENTRY FOR EACH MAP IN THE DISK FILE. THIS ENTRY CONTAINS TWO PIECES OF
0 3 INFORMATION: 1) MAP NUMBER, AND 2) NUMBER OF NODES IN THE MAP. THE
0 4 FIRST ENTRY IN THE MAP DIRECTORY CONTAINS A VALUE EQUAL TO THE NUMBER OF
0 5 MAPS WHICH HAVE BEEN ENTERED. WHEN A MAP'S NODE INFORMATION IS PLACED
0 6 IN THE DATA BASE, ITS DIRECTORY ENTRY IS PLACED IN THE NEXT AVAILABLE
0 7 LOCATION OF THE MAP DIRECTORY. THUS THE SECOND MAP DIRECTORY ENTRY COR-
0 8 RESPONDS TO THE FIRST MAP PLACED IN THE MAPS DATA BASE, THE THIRD ENTRY
0 9 CORRESPONDS TO THE SECOND Map, ETC.

0 0 607
0 0 608
0 0 609
0 0 610
0 0 611
0 0 612
0 0 613
0 0 614
0 0 615

TRW, INC.
21 FEB 78

LOCATION AND MOVEMENT ANALYSIS SYSTEM:
DATA BASE DESCRIPTIONS

PAGE 24

CROSS-COUNTRY DIRECTORY

1 THIS DIRECTORY IS USED TO FIND SPECIFIC MAPS IN THE CROSS-COUNTRY
2 AND CONCEALMENT DISK FILES. EACH MAP HAS ONE DIRECTORY ENTRY, THIS EN-
TRY CONTAINING THE MAP NUMBER, AND THE UTM LATITUDE AND LONGITUDE EX-
TREMES OF THE MAP. THESE ARE THE COORDINATES OF THE LOWER LEFT AND UP-
PER RIGHT CORNERS. THE FIRST ENTRY OF THE DIRECTORY CONTAINS ONLY A
COUNTER WHICH KEEPS TRACK OF HOW MANY ENTRIES ARE PRESENT IN THE DIREC-
TORY.
#

617
618
619
620
621
622
623

TRW, INC.
21 FEB 78

LOCATION AND MOVEMENT ANALYSIS SYSTEM

PAGE 25

* PREPARATION FOR USE *

PREFACE

CREATE NODE+VECTOR AND MAP+DIRECTORY FILES ON DISK

REF PAGE

```
* 1 * BEFORE THE FILES MAY BE CREATED, IT IS NECESSARY TO CREATE 634
* 2 * NODE AND LINK CARDS. INFORMATION FOR THESE CARDS IS VISUALLY 635
* 3 * DETERMINED FROM THE PARTICULAR L-SERIES MAP BEING CONSIDERED. 636
* 4 *
* 5 do FOR EACH MAP 637
* 6   do FOR EACH NODE 638
*    CREATE NODE CARD 639
*   enddo FOR EACH NODE 640
* 7   do FOR EACH LINK 641
*    CREATE LINK CARD 642
*   enddo FOR EACH LINK 643
* 8   enddo FOR EACH MAP 644
* 9 *
*10  .NOW THAT THE CARDS ARE READY FOR PROCESSING, CREATE THE NODE+VECTOR 645
*11 .FILE ON DISK, THAT IS, ESTABLISH NODE+VECTORS USING INFORMATION GAINED 646
*12 .FROM NODE AND LINK CARDS. 647
*13 .FIRST, INITIALIZE THE FILES 648
*14 . 649
*15 .FIRST, INITIALIZE THE FILES 650
*16 . 651
*17 . 652
*18 . 653
*19 do UNTIL 654
*20   CREATE DISK FILE FOR NODE+VECTORS, 'MAPS.DAT'. 655
*21   CREATE DISK FILE FOR MAP+DIRECTORY, 'MAPDIR.DAT'. 656
*22 enddo UNTIL 657
*23 .NOW BUILD THE VECTORS IN MEMORY 658
*24 . 659
*25 .UNTIL ALL MAPS ARE PROCESSED 660
*26   do FOR EACH MAP 661
*27     do UNTIL NO MORE NODE CARDS 662
*28       READ A NODE CARD 663
*29         PLACE INFORMATION INTO MAIN MEMORY 664
*30       enddo UNTIL NO MORE NODE CARDS 665
*31     do UNTIL NO MORE LINK CARDS 666
*32   
```

```
* 33      READ A LINK CARD... NODE A TO NODE B          * 666
* 34      FILL IN LINK INFORMATION TO NODE+VECTOR FOR A    * 667
* 35      IF NODE+VECTOR FOR B IS ON THIS MAP             * 668
* 36          FILL IN LINK INFORMATION TO NODE+VECTOR FOR B    * 669
* 37      ELSE CYCLE                                         * 670
* 38          ENDIF NODE+VECTOR FOR B IS ON THIS MAP        * 671
* 39          ENDDO UNTIL NO MORE LINK CARDS                * 672
* 40          READ "MAPDIR.DAT" INTO MEMORY                 * 673
* 41          READ NUMBER OF MAPS PROCESSED                * 674
* 42          INCREMENT NUMBER OF MAPS PROCESSED           * 675
* 43          CREATE MAP+DIRECTORY ENTRY FOR THIS MAP       * 676
* 44          WRITE NODE+VECTOR ARRAY FOR THIS MAP TO "MAPS.DAT" * 677
* 45          WRITE MAP+DIRECTORY TO "MAPDIR.DAT"           * 678
* 46          ENDDO FOR EACH MAP                           * 679
* 47          ENDDO UNTIL ALL MAPS ARE PROCESSED            * 680
*                                     ****
```

TRW, INC. LOCATION AND MOVEMENT ANALYSIS SYSTEM
21 FEB 78 PREPARATION FOR USE

CREATE NODE CARD

REF PAGE

- * * 1 NODE CARDS WILL BE KEY PUNCHED MANUALLY, USING THE FOLLOWING FORMAT FOR *
- * * 2 THEIR CONSTRUCTION. *
- * * * * *

PAGE 28

NODE CARD FORMAT

#	#	ENTRY	EXAMPLE	FIELD	
# 1	# 1			685	
# 2	# 2			686	
# 3	# 3	NODE MAP NUMBER. THE 1:50000 MAPS WILL BE USED.	4924	1-5(5)	
# 4	# 4	ENTER THE NUMERICAL MAP NUMBER, NOT THE LETTERS.			
# 5	# 5			688	
# 6	# 6			689	
# 7	# 7			690	
# 8	# 8	NODE NUMBER WITHIN THE MAP.	18	6-9(4)	
# 9	# 9	A MASTER OVERLAY (PLASTIC) FOR EACH MAP MUST BE KEPT			
# 10	# 10	# 11 # 12 # 13 # 14 # 15 # 16 # 17 # 18 # 19 # 20 # 21 # 22 # 23 # 24 # 25 # 26 # 27 # 28 # 29 # 30 # 31 # 32 # 33 # 34 # 35 # 36	TO KEEP TRACK OF THE NUMBERS AND NOTES. NODE LATITUDE IN KM'S FROM THE EQUATOR, READ DIRECTLY FROM THE 1:50000 MAP. NODE LONGITUDE IN KM'S FROM GRENNAKH, READ DIRECTLY FROM THE 1:50000 MAP. FIRST ADJACENT NODE MAP #. SECOND ADJACENT NODE MAP, ENTER THE NUMBER ONLY. FIRST ADJACENT NODE NUMBER FOR THE MAP IF THIS ADJACENT NODE. ENTER THE NUMBER ONLY. SECOND ADJACENT NODE DATA, ENTER THE NUMBER ONLY. THIRD ADJACENT NODE DATA, ENTER THE NUMBER ONLY. FOURTH ADJACENT NODE DATA, ENTER THE NUMBER ONLY.	3245 10-16(7) 125 17-23(7) 4923 24-28(5) 201 29-32(4) 706 707 708 709 710 711 712 713 714 715 716 717 718 719 720	693 694 695 696 697 698 699 700 701 702 703 704 705 706 707 708 709 710 711 712 713 714 715 716 717 718 719 720

TRW, INC.
21 FEB 78

LOCATION AND MOVEMENT ANALYSIS SYSTEM
PREPARATION FOR USE

PAGE 29.001

37 NUMBER OF ADJACENT NOTES.

38

#

37 4 60-61(2) 721

38 4 722

#

TRU, INC.
21 FEB 78

LOCATION AND MOVEMENT ANALYSIS SYSTEM
PREPARATION FOR USE

CREATE LINK CARD

REF PAGE

- * 1 * LINK CARDS WILL BE KEY-PUNCHED MANUALLY, USING THE
* 2 * FOLLOWING FORMAT.

PAGE 30

724
725

TRI, INC. LOCATION AND MOVEMENT ANALYSIS SYSTEM
21 FEB 78 PREPARATION FOR USE

LINK CARD FORMAT

PAGE 31

#	ENTRY	EXAMPLE	FIELD	#
# 1			727	728
# 2	FIRST NODE MAP NUMBER.	4924	1-5(5)	729
# 4	FIRST NODE NUMBER WITHIN MAP.	13	6-9(4)	730
# 6	SECOND NODE MAP NUMBER.	4924	10-14(5)	731
# 7	SECOND NODE NUMBER WITHIN MAP.	17	15-18(4)	732
# 8				733
# 9				734
# 10				735
# 11	LINK DISTANCE IN TENTHS OF A KILOMETER BETWEEN FIRST AND SECOND NODES.	18.7	19-23(5)	736
# 12				737
# 13				738
# 14				739
# 15	NUMBER OF EQUIVALENT LANES.	3	24-26(3)	740
# 16				741
# 17	ROAD TYPE CODE.	3	27-28(2)	742
# 18				743
# 19	GROUND COVER CODE.	3		744
# 20				745
# 21	TERRAIN CODE.	2	29-30(2)	746
# 22				747
# 23	BRIDGE CODE.	0	31-32(2)	748
# 24				749
# 25	CITY CODE.	1	33-34(2)	750
# 26				751
#				752

TRU, INC.
21 FEB 78

**LOCATION AND MOVEMENT ANALYSIS SYSTEM
PREPARATION FOR USE**

CREATE MAP+DIRECTORY ENTRY FOR THIS MAP

REF.
PAGE

AT THIS POINT IN THE PROGRAM, THIS MAP'S NODE-VECTORS HAVE BEEN READ INTO CORE, AND ALL PERTINENT INFORMATION HAS BEEN PLACED IN THE PROPER AREA. THE MAP-NUMBER IS IN THE FIRST WORD OF THE NODE-VECTOR ARRAY, AND THE NUMBER OF NODES IN THIS MAP IS KNOWN FROM A COUNTER WHICH WAS INCREMENTED EVERY TIME A NEW NODE CARD WAS READ.

MAP+ DIRECTORY ENTRY(1, NUMBER OF MAPS)=MAP+NUMBER
MAP+ DIRECTORY ENTRY(2, NUMBER+DE+MAPS)=NUMBER DE NOTES

262
754

CREATE CROSS-COUNTRY, CONCEALMENT, AND DIRECTORY FILES ON DISK

REF PAGE

```
* 1 * BEFORE THESE THREE FILES MAY BE ESTABLISHED, THE DATA MUST BE
* 2 .PLACED ON COMPUTER CARDS. THIS DATA IS OBTAINED BY PLACING A
* 3 .93 X 88 GRID OVER THE APPROPRIATE MAP. EACH SQUARE OF THE GRID
* 4 .IS ASSIGNED A VALUE EQUAL TO THE CODE WHICH MOST DESCRIBES THE
* 5 .AREA. THAT IS, IF THE AREA UNDERNEATH THE SQUARE IS PARTIALLY
* 6 .CODE 4, AND PARTIALLY CODE 5, THE VALUE ASSIGNED IS EITHER 4 OR
* 7 .5, DEPENDING UPON WHICH APPEARS TO OCCUPY THE MAJORITY OF THE
* 8 .SQUARE. IF MORE THAN 2 CODES ARE COVERED, THEN THE CODE CHosen
* 9 .IS THE ONE WHICH DOMINATES (SEEMS TO BE GREATER THAN THE OTHERS
* 10 .TAKEN INDIVIDUALLY) THE SQUARE.
* 11 *
* 12 do FOR EACH MAP
* 13   CREATE DATA CARDS
* 14 enddo FOR EACH MAP
* 15 *
* 16 .WITH THE CARDS ALL MADE, IT IS NOW A MATTER OF READING THE CARDS
* 17 .INTO MEMORY, CONVERTING THE VALUES AS APPROPRIATE, MAKING A DIR-
* 18 .ECTORY ENTRY, AND WRITING THE RESULTS OUT TO DISK.
* 19 *
* 20 do DCLCE
* 21   CREATE DISK FILE FOR CROSS-COUNTRY MAPS, "CCM.DAT"
* 22   CREATE DISK FILE FOR WINTER CONCEALMENT MAPS, "WCDN.DAT"
* 23   CREATE DISK FILE FOR SUMMER CONCEALMENT MAPS, "SCDN.DAT"
* 24   CREATE DISK FILE FOR DIRECTORY, "CCMDIR.DAT"
* 25 enddo DCLCE
* 26 *
* 27 .BECAUSE THE ALLOWABLE CODES HAVE THE VALUES 1-7 FOR CROSS-COUN-
* 28 .TRY AND 8-11 FOR CONCEALMENT, ONLY FOUR BITS ARE NEEDED TO STORE
* 29 .THIS INFORMATION. A GOOD SPACE SAVING MAY BE MADE IF THE VALUES
* 30 .ARE PACKED FOUR TO A WORD. DOING THIS, THOUGH, MEANS THAT, IN
* 31 .ORDER TO DEAL WITH COMPLETE ROWS OF DATA, FOUR ROWS OF CARDS
* 32 .SHOULD BE PROCESSED AT ONE TIME.
```

**LOCATION AND MOVEMENT ANALYSIS SYSTEM
PREPARATION FOR USE**

PAGE 33.001

```

* 33 * * do FOR EACH MAP
* 34 * * do until 'N!1. ROWS ARE READ IN
* 35 * * * * * . SINCE EACH ROW HAS 93 ELEMENTS, AND A COMPUTER CARD HAS ONLY
* 36 * * * * * . 72 COLUMNS, EACH ROW USES TWO CARDS.
* 37 * * * * * READ FOUR ROWS WORTH OF CARDS (EIGHT ALTOGETHER)
* 38 * * * * * PACK THESE ROWS FOUR VALUES TO A WORD
* 39 * * * * * enddo UNTIL ALL ROWS ARE READ IN
* 40 * * * * * . ALL DATA IS NOW IN MEMORY. WRITE IT OUT TO PROPER DATA FILE.
* 41 * * * * * WRITE DATA TO DISK
* 42 * * * * * . MAKE DIRECTORY ENTRY
* 43 * * * * * PROMPT FOR MAP+NUMBER
* 44 * * * * * READ INPUT
* 45 * * * * * PLACE INPUT AT NEXT AVAILABLE P_LACE IN COM.DIR ...ONLY DO THIS IF CRESS-COUNTRY MAP
* 46 * * * * * BEING PROCESSED.
* 47 * * * * * PROMPT FOR UTM LATITUDE AND LONGITUDE EXTREMES
* 48 * * * * * PLACE INPUT AT NEXT AVAILABLE PLACE IN COM.DIR ... SAME ADMITION AS BEFORE.
* 49 * * * * * . WRITE DIRECTORY TO DISK
* 50 * * * * * WRITE DIRECTORY TO DISK
* 51 * * * * * enddo FOR EACH MAP

```

TRU, INC.
21 FEB 78

LOCATION AND MOVEMENT ANALYSIS SYSTEM

PAGE: 34

* LAMAS FILE *

TRI, INC.
21 FEB 78

LOCATION AND MOVEMENT ANALYSIS SYSTEM
LAMAS FLOW

LAMAS

REF PAGE

```
*      * 1   do UPON MAIN+PROGRAM EXECUTION          827
*      * 2   do FOREVER
*      * 3   DISPLAY MAIN FUNCTION MENU
*      * 4   WAIT FOR USER INPUT
*      * 5   do CASE OF :
*      * 6   ROAD+NETWORK: IMPLEMENT ALGORITHMS AND FUNCTIONS USING THE NODE+VECTOR DATA BASE    833
*      * 7   TERRAIN+MODEL: IMPLEMENT ALGORITHMS AND FUNCTIONS USING THE CRESS-COUNTRY DATA BASE    834
*      * 8   CONCEALMENT DATA BASES
*      * 9
*      * 10
*      * 11 EXIT: EXIT THIS SYSTEM
*      * 12
*      * 13 enddo CASE OF
*      * 14 enddo FOREVER
*      * 15 enddo UPON MAIN+PROGRAM EXECUTION
*      * 16
*      * 17
*      * 18
*      * 19
*      * 20
*      * 21
*      * 22
*      * 23
*      * 24
*      * 25
*      * 26
*      * 27
*      * 28
*      * 29
*      * 30
*      * 31
*      * 32
*      * 33
*      * 34
*      * 35
*      * 36
*      * 37
*      * 38
*      * 39
*      * 40
*      * 41
```

IMPLEMENT ALGORITHMS AND FUNCTIONS USING THE NODE+VECTOR DATA BASE

REF PAGE

```
* 1 do UPON FUNCTION REQUEST 843
* 2   do FOREVER 844
* 3     DISPLAY FUNCTION MENU 845
* 4     WAIT FOR USER INPUT 846
* 5     do CASE OF 847
* 6   INITIALIZE: 848
* 7     PREPARE FOR ROAD NETWORK PATH CALCULATIONS 849
* 8     CALCULATION: 850
* 9       PATH DETERMINATION AND DISPLAY 851
* 10      EXIT: 852
* 11        return TO CALLING PROGRAM 853
* 12        enddo CASE OF 854
* 13        enddo FOREVER 855
* 14      enddo UPON FUNCTION REQUEST 856
*
```

TRW, INC.
21 FEB 73

LOCATION AND MOVEMENT ANALYSIS SYSTEM
LAMAS FLOW

PAGE 37

PREPARE FOR ROAD NETWORK PATH CALCULATIONS

REF
PAGE

```
* 1 do UPON FUNCTION REQUEST 858
* 2   do FOREVER
* 3     DISPLAY FUNCTION MENU 859
* 4     WAIT FOR USER INPUT 860
* 5       do CASE OF
* 6         MAPS: 861
* 7           INITIALIZE MAP+NUMBERS AND SCREEN EXTREMES 862
* 8           UNITS: 863
* 9             INITIALIZE UNIT+VECTORS 864
* 10            FERO: 865
* 11             ESTABLISH A FORWARD EDGE OF BATTLE AREA 866
* 12             EXIT: 867
* 13               return TO CALLING PROGRAM 868
* 14             enddo CASE OF 869
* 15             enddo FOREVER 870
* 16             enddo UPON FUNCTION REQUEST 871
* 872
* 873
```

TRI, INC.
21 FEB 78

LOCATION AND MOVEMENT ANALYSIS SYSTEM
LAMAS FLIN

INITIALIZE MAP-NUMBERS AND SCREEN EXTREMES

REF
PAGE

```
*      * 1 do UPON FUNCTION REQUEST
*      * 2   ESTABLISH DIRECTORY IN MEMORY AND READ IN NODE-VECTORS
*      * 3   do FOR EACH NODE IN MEMORY
*      * 4     do FOR EACH ADJACENT NODE
*             if ADJACENT NODE'S MAP-NUMBER IS IN THE DIRECTORY
*               CHANGE ADJACENT NODE'S MAP-NUMBER TO INDEX
*             endif ADJACENT NODE'S MAP-NUMBER
*             enddo FOR EACH ADJACENT NODE
*      * 5
*      * 6
*      * 7
*      * 8
*      * 9
*      * 10
*      * 11
*      * 12
*      * 13
*      * 14
```

875 * 876 * 877 * 878 * 879 * 880 * 881 * 882 * 883 * 884 * 885 * 886 * 887 * 888

PAGE 38

ESTABLISH DIRECTORY IN MEMORY AND READ IN NODE+VECTORS

REF PAGE

```
* 1   THIS DIRECTORY CONSISTS OF A NUMBER OF ENTRIES, ONE FOR EACH
* 2   MAP WHICH THE USER WISHES TO EMPLOY. THE FIRST DIRECTORY ENTRY CORRE-
* 3   SPONDS TO THE FIRST MAP+NUMBER REQUESTED BY THE USER, AND SO ON FOR
* 4   EACH ENTRY. THERE ARE THREE ELEMENTS IN EACH ENTRY, 1) THE MAP+NUMBER,
* 5   2) THE NUMBER OF NODES IN THE MAP, AND 3) THE NUMBER OF NODES ALREADY
* 6   PRESENT IN MEMORY.
* 7   GIVEN THIS MAP+DIRECTORY, IT IS A SIMPLE MATTER TO READ A MAP'S
* 8   NODES INTO MAIN MEMORY, SINCE WE KNOW HOW MANY TO READ IN,
* 9   MAP+DIRECTORY WORD 2, AND WHERE TO START PUTTING THEM, MAP+DIRECTORY
* 10  .WORD 3 +1.
* 11
* 12  READ DISK FILE MAP+DIRECTORY INTO MAIN MEMORY
* 13  PROMPT FOR ENTRY OF MAP+NUMBERS
* 14  READ USER INPUT OF MAP+NUMBERS TO BE PROCESSED
* 15  DO FOR EACH MAP+NUMBER ENTERED BY USER
* 16  IF MAP+NUMBER IS IN THE DISK FILE MAP+DIRECTORY...AS THE 1 TH ENTRY
* 17  SET FIRST WORD OF MAIN MEMORY MAP+DIRECTORY TO :
* 18  SET SECOND WORD TO NUMBER OF NODES IN THIS MAP
* 19  SET THIRD WORD TO NUMBER OF NODES ALREADY PRESENT IN MEMORY
* 20  READ THIS MAP'S NODES INTO MAIN MEMORY
* 21  RESET FIRST WORD TO THIS MAP+NUMBER
* 22  ELSE MAP+NUMBER IS NOT IN THE DATA BASE
* 23  PRINT ERROR MESSAGE
* 24  LET THE USER RE-ENTER THE MAP+NUMBER
* 25  cycle
* 26  endif MAP+NUMBER IS IN THE DISK FILE MAP+DIRECTORY
* 27  enddo FOR EACH MAP+NUMBER ENTERED BY USER
```

REND THIS MAP'S NODES INTO MAIN MEMORY

REF:
PAGE:

```
* 1 * AT THIS POINT IN THE PROGRAM, THE MAP+ DIRECTORY IN MAIN MEMORY
* 2 * HAS ITS AS ELEMENTS 1) THE INDEX 1, INDICATING THAT THIS MAP IS THE
* 3 * 1 TH ONE IN THE DISK FILE MAP+ DIRECTORY, 2) NUMBER OF NODES IN THIS MAP
* 4 * AND 3) NUMBER OF NODES ALREADY PRESENT IN MEMORY. IN ORDER TO
* 5 * READ THIS MAP'S NODES INTO MEMORY SO THAT NO SPACE IS WASTED,
* 6 * WE NEED ONLY CONSIDER THE INFORMATION JUST LISTED.
* 7 * FROM DISK TO MEMORY WE NEED TO KNOW HOW MANY BYTES WILL BE MOVED.
* 8 * THIS IS JUST WORD 2 MULTIPLIED BY 64 (32 RECORDS * 2 BYTES PER WORD). HE
* 9 * ALSO MUST KNOW EXACTLY WHERE THE DATA WILL BE PLACED IN MEMORY, THAT
* 10 * IS, THE NEXT AVAILABLE SPACE IN THE NODE+VECTOR ARRAY.
* 11 * THIS IS SIMPLY WORD 3 PLUS ONE. FINALLY, WE MUST KNOW FROM THAT
* 12 * DISK BLOCK IS THE DATA COMING, THIS IS I-1 MULTIPLIED BY 10, SINCE
* 13 * THE FIRST MAP STARTS AT BLOCK ONE, THE SECOND AT BLOCK 11, ETC.
* 14 * NOTE THAT NO MAP HAS MORE THAN 80 NODE+ VECTORS, EACH VECTOR
* 15 * HAVING 32 WORDS, SO THERE ARE 8 NODE+ VECTORS TO A DISK BLOCK OF
* 16 * 256 WORDS.
* 17 *
* 18 * BYTES = (MAP+ DIRECTORY WORD 2) * 64
* 19 * INDEX = (MAP+ DIRECTORY WORD 3) + 1
* 20 * STARTING BLOCK = (MAP+ DIRECTORY WORD 1) - 1; * 10
* 21 * PERFORM DISK READ (BYTES, INDEX, STARTING BLOCK)
* 22 *
```

TRW, INC.
21 FEB 78

LOCATION AND MOVEMENT ANALYSIS SYSTEM
LAMAS FLD:

CHANGE ADJACENT NODE'S MAP-NUMBER TO INDEX

REF
PAGE

* 1 * IN ORDER TO ACCESS A NODE QUICKLY, IT IS NECESSARY THAT ONE NOT
* 2 * HAVE TO USE THE MAP+DIRECTORY EACH TIME A REFERENCE IS MADE.
* 3 * THUS, CONSIDER THE NODE+VECTOR ARRAY AS A LIST OF NODES,
* 4 * INDEXABLE BY INTEGERS, SUCH THAT THE FIRST NODE+VECTOR HAS INDEX=1,
* 5 * THE SECOND HAS INDEX=2, AND SO ON FOR THE ENTIRE LIST. THEN, IF
* 6 * WE KNOW THE MAP+NUMBER AND THE NODE+NUMBER OF A PARTICULAR NODE+VECTOR,
* 7 * THEN THIS INDEX MAY BE CALCULATED ONCE, AND SAVED. THE MOST LOGICAL
* 8 * PLACE TO SAVE IT IS IN IT'S ADJACENT MAP+NUMBER LOCATION.
* 9 * THIS CALCULATION IS SIMPLE, AND DEPENDS ON OUR KNOWLEDGE OF JUST
* 10 * TWO THINGS, 1) FOR THE ADJACENT NODE'S MAP+NUMBER, THE NUMBER OF NODES
* 11 * ALREADY IN CORE, AND 2) FOR THE ADJACENT NODE, ITS NODE+NUMBER. BOTH
* 12 * OF THESE PIECES OF INFORMATION ARE READILY AVAILABLE, THE FIRST IS JUST
* 13 * THE THIRD WORD OF THE ADJACENT NODE'S MAP+DIRECTORY ENTRY, AND
* 14 * THE SECOND IS IN THE NODE+VECTOR.
* 15 *
* 16 * ADJACENT MAP+NUMBER = (ADJACENT NODE'S MAP+DIRECTORY WORD 3) + ADJACENT INDEX.

940
941
942
943
944
945
946
947
948
949
950
951
952
953
954

INITIALIZE UNIT+VECTORS

REF
PANIC

```
*****  
* 1   * THIS DATA BASE IS COMPLETELY USER DEFINED AT EXECUTION TIME.  
* 2   * THE USER SPECIFIES UNIT NAME, UNIT STARTING AND DESTINATION LO-  
* 3   * CATIONS, UNIT PRIORITY, AND UNIT TYPE CODE.  
* 4   *  
* 5   * DO UPON FUNCTION REQUEST  
* 6   * ASK IF USER WISHES TO REINITIALIZE OR ADD ON TO EXISTING STRUCTURE  
* 7   * IF USER WANTS TO REINITIALIZE  
* 8       * ERASE ALL UNIT+VECTORS  
* 9       * SET UNIT+COUNTER TO ZERO  
* 10      *endif user wants to reinitialize  
* 11      *  
* 12      *PERFORM THE FOLLOWING AS MANY AS SIXTY TIMES  
* 13      *  
* 14      *do UNTIL UNIT+COUNTER EQUALS SIXTY  
* 15      *  
* 16      *PROMPT FOR THE FOLLOWING: UNIT NAME, STARTING LOCATION, DESTINATION,  
* 17      *PRIORITY, AND TYPE CODE. ERROR CHECKING IS BE PERFORMED ON ALL INPUT  
* 18      *SAVE THE UNIT NAME. INPUT FOR THE START AND FINISH LOCATIONS IS IN THE  
* 19      *FORM MAP+NUMBER, NODE+NUMBER. ANY TIME THERE IS AN ERROR, THE PROMPT  
* 20      *WILL BE REPEATED UNTIL VALID INPUT IS RECEIVED.  
* 21      *  
* 22      *PROMPT USER TO ENTER UNIT NAME  
* 23      *if UNIT NAME IS NULL  
* 24          *return TO CALLING PROGRAM  
* 25      *else INPUT IS VALID CHARACTERS  
* 26          *INCREMENT UNIT+COUNTER  
* 27          *ASSIGN INPUT TO FIRST FOUR WORDS OF NEXT AVAILABLE UNIT+VECTOR  
* 28          *PROMPT FOR STARTING LOCATION  
* 29          *if STARTING LOCATION IS INVALID  
* 30              *PRINT ERROR MESSAGE  
* 31              *TRY AGAIN  
* 32          *else INPUT IS VALID
```

958 959 960
961 962 963
964 965 966
967 968 969
970 971 972
973 974 975
976 977 978
979 980 981
982 983 984
985 986 987
988 989 989

TRW, INC.
21 FEB 78

LOCATION AND MIMENT ANALYSIS SYSTEM
LAMAS FILE:

PAGE 42.001

```
* 33      ASSIGN INPUT TO UNIT+VECTOR WORD 5 (STARTING LOCATION)
* 34      ASSIGN INPUT TO UNIT+VECTOR WORD 7 (PRESENT LOCATION)
* 35      PROMPT FOR DESTINATION
* 36      if DESTINATION IS INVALID
* 37          PRINT ERROR MESSAGE
* 38          TRY AGAIN
* 39      else INPUT IS VALID
* 40          ASSIGN INPUT TO UNIT+VECTOR WORD 6 (DESTINATION)
* 41          PROMPT FOR PRIORITY
* 42          if PRIORITY IS INVALID
* 43              PRINT ERROR MESSAGE
* 44          TRY AGAIN
* 45      else INPUT IS VALID
* 46          ASSIGN INPUT TO FIRST BYTE OF UNIT+VECTOR WORD 3
* 47          PROMPT FOR TYPE CODE
* 48          if TYPE CODE IS INVALID
* 49              PRINT ERROR MESSAGE
* 50          TRY AGAIN
* 51      else INPUT IS VALID
* 52          ASSIGN INPUT TO SECOND BYTE OF UNIT+VECTOR WORD 8
* 53          endif TYPE CODE IS INVALID
* 54          endif PRIORITY IS INVALID
* 55          endif DESTINATION IS INVALID
* 56          endif STARTING LOCATION IS INVALID
* 57          endif UNIT NAME IS NULL
* 58          enddo UNIT+COUNTER EQUALS SIXTY
* 59          enddo UPON FUNCTION REQUEST
* 60
```

ESTABLISH A FORWARD EDGE OF BATTLE AREA

REF PAGE

```
* 1   *AN ARBITRARY MAXIMUM OF SIX NODES MAY BE DESIGNATED AS FEBA POINTS.      *
* 2   *                                             *1018
* 3   do UPON FUNCTION REQUEST          *1019
* 4   do SIX TIMES                      *1020
* 5   PROMPT FOR MAP+NUMBER AND NODE+NUMBER, CR EXIT      *1021
* 6   if INPUT IS EXIT                  *1022
* 7   return TO CALLING PROGRAM        *1023
* 8   endif INPUT IS EXIT              *1024
* 9   CHECK MAP+NUMBER AND NODE+NUMBER FOR LEGITIMACY    *1025
* 10  if NOT VALID                   *1026
* 11  PRINT ERROR MESSAGE            *1027
* 12  TRY AGAIN                     *1028
* 13  else VALID                    *1029
* 14  FIND NODE+VECTOR CORRESPONDING TO MAP+NUMBER, NODE+NUMBER      *1030
* 15  ASSIGN NODE+VECTOR TO FEBA ARRAY          *1031
* 16  endif NOT VALID                *1032
* 17  enddo SIX TIMES                 *1033
* 18  enddo UPDN FUNCTION REQUEST      *1034
*                                         *1035
```

TRW, INC.
21 FEB 78

LOCATION AND MOVEMENT ANALYSIS SYSTEM
LAMAS FLOW

PAGE 44

FIND NODE+VECTOR NEAREST TO UTM COORDINATES

KEY
P&HST

```
* 1 .THIS WILL BE ACCOMPLISHED BY STARTING AT THE BEGINNING OF THE
* 2 .NODE+VECTOR ARRAY AND COMPUTING EACH NODE+VECTOR'S DISTANCE FROM THE
* 3 .UTM COORDINATES USING A LEAST SQUARES APPROACH. THAT IS, DISTANCE IS
* 4 .CALCULATED TO BE THE SUM OF THE SQUARES OF THE DIFFERENCES BETWEEN THE
* 5 .NODE+VECTOR 'S LATITUDE AND LONGITUDE AND THE CALCULATED UTM COORDI-
* 6 .NATES.
* 7 .
* 8 .LEAST = INFINITY
* 9 .do FOR EACH NODE+VECTOR
* 10 .DISTANCE = <UTM LAT. - LAT.> ** 2 + <UTM LON. - LON.> ** 2
* 11 .if DISTANCE IS LESS THAN LEAST
* 12 .REMEMBER THE NODE+VECTOR
* 13 .LEAST = DISTANCE
* 14 .endif DISTANCE IS LESS THAN LEAST
* 15 .enddo FOR EACH NODE+VECTOR
* 16 .
* 17 .REMEMBERED NODE+VECTOR IS THE ONE WE WANT
* 18 .
* 19 .
```

PATH DETERMINATION AND DISPLAY

REF PAGE

```
* 1 * THESE ARE THE FUNCTIONS WHICH PERFORM THE FUNDAMENTAL MOVEMENT ALGO-
* 2 * -RITHM. THE MAIN PURPOSE IS TO COMPUTE OPTIMAL PATHS FOR DEFINED
* 3 * GROUND FORCE UNITS, SO THAT CERTAIN CRITERIA IS SATISFIED. THE GENERAL
* 4 * PROCEDURE IS, GIVEN A UNIT'S STARTING NODE, DESTINATION NODE, AND
* 5 * DESIRED ARRIVAL TIME AT DESTINATION, OR START TIME AT THE BEGINNING,
* 6 * CALCULATE THE BEST PATH FOR THE UNIT TO TRAVEL. "BEST" IS USER-DEFINED
* 7 * AS A FUNCTION OF TIME AND RISK. THE FORMULA USED IS  $K*TIME + C*RISK$ ,
* 8 * WHERE K AND C ARE CONSTANTS, C EQUALING ZERO OR ONE, AND K BEING BE-
* 9 * TWEEN ZERO AND TEN. BOTH K AND C MAY NOT BE ZERO AT THE SAME TIME.
* 10 * USING THE BASIC PATH ALGORITHM, CERTAIN VARIATIONS OF THE PATH CALCUL-
* 11 * ATION MAY BE PERFORMED, SUCH AS FINDING THE SECOND SHORTEST PATH,
* 12 * FINDING THE OPTIMAL NODE FOR INTERDICTION, AND, GIVEN A NUMBER OF UNITS
* 13 * TRAVELING THROUGH A ROAD NETWORK, FIND WHICH ORDER OF PRIORITIES OPTI-
* 14 * MIZES THE ENTIRE NETWORKS USAGE.
* 15 * THERE ARE TWO MORE OPTIONS FROM WHICH THE USER MAY CHOOSE: INTERDIC-
* 16 * TION AND SOLUTIONS. THEIR PROPERTIES WILL BE EXPLAINED LATER.
* 17 *
* 18 do UPON FUNCTION REQUEST
* 19   do FOREVER
* 20     DISPLAY FUNCTION MENU
* 21     WAIT FOR USER INPUT
* 22     do CASE IF
* 23       INTERDICTION:    PERFORM INTERDICTIVE OPERATIONS
* 24
* 25       PATH:          1079
* 26         CALCULATE PATH(S) AND SOLUTION VECTORS
* 27       SECOND:        1080
* 28         CALCULATE SECOND BEST PATH
* 29       NETWORK:      1081
* 30         FIND THE ORDER OF PRIORITIES FOR BEST NETWORK TRAVEL
* 31       BEST:          1082
* 32         CALCULATE BEST NODE AT WHICH TO INTERDICT
* 33
* 34
* 35
* 36
* 37
* 38
* 39
* 40
* 41
* 42
* 43
* 44
* 45
* 46
* 47
* 48
* 49
* 50
* 51
* 52
* 53
* 54
* 55
* 56
* 57
* 58
* 59
* 60
* 61
* 62
* 63
* 64
* 65
* 66
* 67
* 68
* 69
* 70
* 71
* 72
* 73
* 74
* 75
* 76
* 77
* 78
* 79
* 80
* 81
* 82
* 83
* 84
* 85
* 86
* 87
```

TRK, INC.
21 FEB 78

LOCATION AND MOVEMENT ANALYSIS SYSTEM
LAMAS FLOW

PAGE 45.001

```
* 33 SOLUTION:  
63 * 34 PRESENT RESULTS  
* 35 EXIT:  
* 36      return TO CALLING PROGRAM  
* 37      enddo CASE OF  
* 38      enddo FOREVER  
* 39      enddo UPDN FUNCTION REQUEST  
*
```

```
* 1088  
* 1089  
* 1090  
* 1091  
* 1092  
* 1093  
* 1094  
*
```

TRW, INC.
21 FEB 78

LOCATION AND MOVEMENT ANALYSIS SYSTEM
LAMAS FLOW:

PAGE 46

PERFORM INTERDICITIVE OPERATIONS

REF
PAGE

```
*****  
* 1   HERE, THE USER MAY INTERDICT TO FORCE UNITS TO MOVE A CERTAIN      *  
* 2   WAY, TO KAKE LINKS IMPossible, OR TO SEE THE EFFECTS OF SUCH INTERDIC-  *  
* 3   TION. THE USER MAY ALSO LOOK AT THE CONTENTS OF NODE+VECTORS.          *  
* 4   *****  
* 5   do UPON FUNCTION REQUEST                                         1096  
* 6   do FOREVER                                                 1097  
* 7   DISPLAY FUNCTION MENU                                         1098  
* 8   WAIT FOR USER INPUT                                         1099  
* 9   do CASE OF  
* 10 VECTORE:  
* 11   PRINT MAP+NUMBER, NODE+NUMBER OF NODE+VECTOR NEAREST GIVEN COORDINATES  
* 12 CONTENTS:                                              1100  
* 13   CHANGE CONTENTS OF A NODE+VECTOR                           1101  
* 14 ERASE:                                                 1102  
* 15   DELETE A NODE FROM THE NODE+VECTOR ARRAY                 1103  
* 16 SHIJ:                                                 1104  
* 17   PRINT CONTENTS OF A NODE+VECTOR                           1105  
* 18 PROPERTIES:                                             1106  
* 19   CHANGE PROPERTIES OF A UNIT                            1107  
* 20 EXIT:                                                 1108  
* 21   return TO CALLING PROGRAM                         1109  
* 22   enddo CASE OF                                         1110  
* 23   enddo FOREVER                                         1111  
* 24   enddo UPON FUNCTION REQUEST                         1112  
* 1113  
* 1114  
* 1115  
* 1116  
* 1117  
* 1118  
* 1119  
*****
```

PRINT MAP-NUMBER, NODE-NUMBER OF NODE+VECTOR NEAREST GIVEN COORDINATES

REF PAGE

```
* 1 * THIS ALLOWS THE USER TO FIND OUT WHICH NODE+VECTOR IS NEAREST ANY UTM
* 2 * COORDINATES OF INTEREST. THIS IS OFTEN HELPFUL IF THE TRACKBALL IS
* 3 * UNAVAILABLE.
*
* 4 * PROMPT FOR USER INPUT UTM COORDINATES
* 5 * SET LEAST TO INFINITY
* 6 * do FOR EACH NODE+VECTOR
* 7 *   DISTANCE = (UTM LAT.-LATITUDE) ** 2 + (UTM LON. - LONGITUDE) **2
* 8 *   if DISTANCE IS LESS THAN LEAST
* 9 *     SET LEAST EQUAL TO DISTANCE
* 10 *     REMEMBER NODE+VECTOR
* 11 *   endif DISTANCE IS LESS THAN LEAST
* 12 * enddo FOR EACH NODE+VECTOR
* 13 *
* 14 *   THE ANSWER IS THE LAST REMEMBERED NODE+VECTOR
* 15 *
* 16 *   PRINT OUT MAP-NUMBER, NODE-NUMBER OF NODE+VECTOR
* 17 *
* 18 *
```

CHANGE CONTENTS OF A NODE+VECTOR

REF
PAGE

```
* 1   THIS FUNCTION ALLOWS THE USER TO CHANGE CERTAIN CONDITIONS OF ANY
* 2   NODE+VECTOR. THE USER MAY CHANGE ANY OF SIX LINK CHARACTERISTICS,
* 3   NUMBER OF LANES, LINK DISTANCE, TERRAIN CODE, BRIDGE CODE, CITY CODE,
* 4   OR ROAD TYPE.
*
* 5   do UPON FUNCTION REQUEST
* 6   PROMPT ENTRY OF MAP+NUMBER, NODE+NUMBER
* 7   if ENTERED NODE+VECTOR IS NOT IN MEMORY
* 8       PRINT ERROR MESSAGE
* 9
* 10      TRY AGAIN
* 11
* 12      else ENTERED NODE+VECTOR IS IN MEMORY
* 13          PROMPT FOR LINKING MAP+NUMBER, NODE+NUMBER
* 14          if LINKING NODE+VECTOR IS NOT IN MEMORY
* 15              PRINT ERROR MESSAGE
* 16              TRY AGAIN
* 17          else LINKING NODE+VECTOR IS IN MEMORY
* 18              if LINKING NODE+VECTOR DOES NOT LINK TO FIRST ENTERED NODE+VECTOR
* 19                  PRINT ERROR MESSAGE
* 20                  TRY AGAIN
* 21                  else TWO NAMED NODE+VECTORS ARE IN MEMORY AND ARE LINKED
* 22                      PRINT LIST OF CHOICES FOR ALTERATION
* 23                      WAIT FOR USER INPUT
* 24                      do CASE OF
* 25
* 26      CHANGE CHARACTERISTIC ... ROAD TYPE
* 27
* 28      CHANGE CHARACTERISTIC ... LINK DISTANCE
* 29
* 30      CHANGE CHARACTERISTIC ... TERRAIN CODE
* 31
* 32      CHANGE CHARACTERISTIC ... NUMBER OF LANES
* 33
* 34      CHANGE CHARACTERISTIC ... BRIDGE CODE
*
* 35
* 36
* 37
* 38
* 39
* 40
* 41
* 42
* 43
* 44
* 45
* 46
* 47
* 48
* 49
* 50
* 51
* 52
* 53
* 54
* 55
* 56
* 57
* 58
* 59
* 60
* 61
* 62
* 63
* 64
* 65
* 66
* 67
* 68
* 69
* 70
* 71
* 72
* 73
* 74
* 75
* 76
* 77
* 78
* 79
* 80
* 81
* 82
* 83
* 84
* 85
* 86
* 87
* 88
* 89
* 90
* 91
* 92
* 93
* 94
* 95
* 96
* 97
* 98
* 99
* 100
* 101
* 102
* 103
* 104
* 105
* 106
* 107
* 108
* 109
* 110
* 111
* 112
* 113
* 114
* 115
* 116
* 117
* 118
* 119
* 120
* 121
* 122
* 123
* 124
* 125
* 126
* 127
* 128
* 129
* 130
* 131
* 132
* 133
* 134
* 135
* 136
* 137
* 138
* 139
* 140
* 141
* 142
* 143
* 144
* 145
* 146
* 147
* 148
* 149
* 150
* 151
* 152
* 153
* 154
* 155
* 156
* 157
* 158
* 159
* 160
* 161
* 162
* 163
* 164
* 165
* 166
* 167
* 168
* 169
* 170
```

TRN, INC.
21 FEB 78

LOCATION AND MOVEMENT ANALYSIS SYSTEM
LAMAS FLM

PAGE 48.001

```
49 * 33          CHANGE CHARACTERISTIC ...BRIDGE CODE
    * 34 CITY:      CHANGE CHARACTERISTIC ...CITY CODE
    49 * 35          enddo CASE OF
    * 36          return TO CALLING PROGRAM
    * 37          endif LINKING NODE+VECTOR DOES NOT LINK TO FIRST ENTERED NODE+VECTOR
    * 38          endif LINKING NODE+VECTOR IS NOT IN MEMORY
    * 39          endif ENTERED NODE+VECTOR IS NOT IN MEMORY
    * 40          enddo UPON FUNCTION REQUEST
    * 41          *
    * 42          *
    * 43          *
    * 44          *
    * 45          *
    * 46          *
    * 47          *
    * 48          *
    * 49          *
    * 50          *
    * 51          *
    * 52          *
    * 53          *
    * 54          *
    * 55          *
    * 56          *
    * 57          *
    * 58          *
    * 59          *
    * 60          *
    * 61          *
    * 62          *
    * 63          *
    * 64          *
    * 65          *
    * 66          *
    * 67          *
    * 68          *
    * 69          *
    * 70          *
    * 71          *
    * 72          *
    * 73          *
    * 74          *
    * 75          *
    * 76          *
    * 77          *
    * 78          *
    * 79          *
```

CHANGE CHARACTERISTIC

REF

PAGE *****

```
* 1   * THIS IS THE SAME PROCEDURE REGARDLESS OF WHICH CHARACTERISTIC IS BEING
* 2   .CHANGED.
* 3
* 4   PRINT OUT OLD VALUE
* 5   PROMPT FOR NEW VALUE
* 6   IF NEW VALUE WITHIN ACCEPTABLE LIMITS
* 7       ASSIGN NEW VALUE TO NODE VECTOR
* 8   ELSE NEW VALUE IS NOT ACCEPTABLE
* 9   PRINT ERROR MESSAGE
* 10  TRY AGAIN
* 11  ENDIF NEW VALUE IS WITHIN ACCEPTABLE LIMITS
* 12
* 13
```

* 1181
* 1182
* 1183
* 1184
* 1185
* 1186
* 1187
* 1188
* 1189
* 1190
* 1191

DELETE A NODE+VECTOR ARRAY

REF PAGE

```
*      1   FOR THE PURPOSE OF INTERDICTION IT MAY BE DESIRED THAT A NODE+VECTOR
*      2   .BE 'ERASED', THAT IS, AFFECTED SO THAT NO UNIT WILL TRAVEL THROUGH IT.
*      3   .THIS ROUTINE CHANGES THE NUMBER OF LANES PARAMETER FOR EACH ADJACENT
*      4   .LINK TO 0. IF THERE'S NO ROAD, NO UNIT CAN TRAVEL THERE. ONCE
*      5   .DONE, RESTORATION IS PERFORMED BY EITHER REESTABLISHING EACH LINK
*      6   .MANUALLY, OR BY READING THE MAPS FROM DISK AGAIN.
*
*      7   do UPON FUNCTION REQUEST
*      8   PROMPT FOR MAP-NUMBER, NODE+NUMBER OF NODE+VECTOR TO BE ERASED
*      9   if ENTERED VALUES ARE INVALID
*     10    PRINT ERROR MESSAGE
*     11    TRY AGAIN
*     12    else ENTERED VALUES ARE VALID
*     13    do FOR EACH ADJACENT NODE
*     14      SET NUMBER OF LANES TO THIS NODE TO ZERO
*     15      enddo FOR EACH ADJACENT NODE
*     16      return TO CALLING PROGRAM
*     17    endif ENTERED VALUES ARE INVALID
*     18    enddo UPON FUNCTION REQUEST
*     19
```

1193 * 1194 *
1194 * 1195 *
1195 * 1196 *
1196 * 1197 *
1197 * 1198 *
1198 * 1199 *
1199 * 1200 *
1200 * 1201 *
1201 * 1202 *
1202 * 1203 *
1203 * 1204 *
1204 * 1205 *
1205 * 1206 *
1206 * 1207 *
1207 * 1208 *
1208 * 1209 *
1209 * 1210 *
1210 * 1211 *

PRINT CONTENTS OF A NODE+VECTOR

REF PAGE

```
* 1 do UPON FUNCTION REQUEST
* 2   PROMPT FOR MAP+NUMBER, NODE+NUMBER
* 3   if ENTERED VALUES ARE NOT VALID
* 4     PRINT ERROR MESSAGE
* 5     TRY AGAIN
* 6   else ENTERED VALUES ARE VALID
* 7     PRINT CONTENTS OF NODE+VECTOR
* 8     return TO CALLING PROGRAM
* 9   endif ENTERED VALUES ARE NOT VALID
* 10 enddo UPON FUNCTION REQUEST
```

1213
* * *
1214
* * *
1215
* * *
1216
* * *
1217
* * *
1218
* * *
1219
* * *
1220
* * *
1221
* * *
1222

CHANGE PROPERTIES OF A UNIT

REF
PAGE

```
*****  
* 1   AT THIS TIME, THE ONLY PROPERTY WHICH WE WILL ALLOW TO BE CHANGED,  
* 2   IS THE UNIT'S PRIORITY.  
* 3  
* 4   do UPON FUNCTION REQUEST  
* 5     PROMPT ENTRY OF UNIT NAME  
* 6     if INPUT NAME HAS AN ENTRY IN UNITS FILE  
* 7       PRINT OLD VALUE  
* 8       PROMPT ENTRY OF NEW PRIORITY  
* 9       if INPUT IS BETWEEN 0 AND 127  
* 10      SET NEW VALUE IN UNIT<VECTOR  
* 11      return TO CALLING PROGRAM  
* 12     else INPUT IS NOT VALID  
* 13       PRINT ERROR MESSAGE  
* 14       TRY AGAIN  
* 15     endif INPUT IS BETWEEN 0 AND 127  
* 16     else INPUT NAME HAS NO ENTRY IN UNITS FILE  
* 17       PRINT ERROR MESSAGE  
* 18       TRY AGAIN  
* 19     endif INPUT NAME HAS AN ENTRY IN UNITS FILE  
* 20   enddo UPON FUNCTION REQUEST  
*****
```

CALCULATE PATH(S) AND SOLUTION VECTORS

REF PAGE

1 THIS ROUTINE WILL DETERMINE THE BEST PATH FOR GROUND FORCE
2 UNITS TO USE, TAKING INTO CONSIDERATION TRAVEL TIME, GROUND COVER, TER-
3 RAIN, BRIDGES, AND CITIES.
4 BEST DOESN'T ONLY IMPLY FASTEST, BUT MAY MEAN LEAST RISKY, OR SOME
5 COMBINATION OF SPEED AND RISK.
6 THERE ARE SEVERAL OPTIONS FROM WHICH THE USER IS FREE TO CHOOSE. AMONG
7 THESE ARE:
8 1) SIMPLE BEST PATH CALCULATION, REGARDLESS OF TIME CONFLICTS,
9 2) AUTOMATIC CONFLICT RESOLUTION; THE PROGRAM DOES THE RESOLUTION
10 WITHOUT USER INTERVENTION.
11 3) WORK FORWARD IN TIME, THAT IS, GIVE A STARTING TIME AND NODE,
12 AND A DESTINATION NODE. THE PROGRAM WILL THEN WORK FORWARD IN TIME,
13 BEGINNING AT THE START NODE, AND ENDING AT THE DESTINATION NODE.
14 4) WORK BACKWARDS IN TIME. GIVEN A STARTING NODE, DESTINATION NODE
15 AND ARRIVAL TIME, START AT THE DESTINATION NODE, WORK BACKWARDS IN TIME
16 TO THE STARTING NODE.
17 AFTER THE PATH HAS BEEN DETERMINED, THE ROUTE IS RECONSTRUCTED USING
18 SOLUTION-VECTORS. THESE VECTORS RECORD THE TIMES DURING WHICH THE VAR-
19 IUS NODES ARE BUSY. THIS INFORMATION WILL BE USED WHEN RESOLVING
20 TIME CONFLICTS AMONG THE PATHS.
21
22 do UPON FUNCTION REQUEST
23 .BEFORE ANYTHING ELSE, CLEAR AWAY ALL OLD ROUTES AND SOLUTIONS
24 PURGE ALL ROUTES
25 .NOW GET PATH PARAMETERS
26
27 PROMPT FOR ENTRY OF MOVEMENT TYPE (BACKWARD OR FORWARD IN TIME)
28 PROMPT FOR ENTRY OF TIME CONFLICT RESOLUTION CODE (NONE OR AUTOMATIC)
29
30
31
32

```
* 33 * HAVE USER ENTER C AND K, THE COEFFICIENTS FOR RISK AND TIME
* 34 * PROMPT FOR ENTRY OF C
* 35 * IF C NOT EQUAL 10 0 OR 1
* 36 * PRINT MESSAGE
* 37 * TRY AGAIN
* 38 * ENDIF C NOT EQUAL TO 0 OR 1
* 39 * PROMPT FOR ENTRY OF K
* 40 * IF K NOT BETWEEN 0 AND 9.99
* 41 * PRINT ERROR MESSAGE
* 42 * TRY AGAIN
* 43 * ENDIF K NOT BETWEEN 0 AND 9.99
* 44 * IF C AND K EQUAL 0
* 45 * PRINT ERROR MESSAGE
* 46 * TRY AGAIN
* 47 * ENDIF C AND K EQUAL 0
* 48 *
* 49 * PROMPT FOR UNIT NAMES TO BE CONSIDERED
* 50 *
* 51 * DO UNTIL USER ENTERS NULL
* 52 * PROMPT FOR ENTRY OF UNIT NAME
* 53 * IF INPUT IS NULL
* 54 * UNDO UNTIL USER ENTERS NULL
* 55 * ELSE INPUT IS NOT NULL
* 56 * REMEMBER UNIT
* 57 *
* 58 * ENDIF INPUT IS NULL
* 59 * UNDO UNTIL USER ENTERS NULL
* 60 *
* 61 * CALCULATE THE PATHS
* 62 *
* 63 * DO FOR EACH ENTERED UNIT, IN ORDER OF PRIORITY
* 64 * IF MOVEMENT TYPE IS BACKWARDS
* 65 * PROMPT FOR ARRIVAL TIME
* 66 * ELSE MOVEMENT TYPE IS FORWARDS
* 67 * PROMPT FOR LEAVING TIME
* 68 * ENDIF MOVEMENT TYPE IS BACKWARDS
* 69 *
* 70 * THE "WORKING LIST" IS A FICTIONAL ARRAY WHICH KEEPS TRACK OF VARIOUS
* 71 * 1277 * 1278 * 1279 * 1280 * 1281 * 1282 * 1283 * 1284 * 1285 * 1286 * 1287 * 1288 * 1289 * 1290 * 1291 * 1292 * 1293 * 1294 * 1295 * 1296 * 1297 * 1298 * 1299 * 1300 * 1301 * 1302 * 1303 * 1304 * 1305 * 1306 * 1307 * 1308 * 1309 * 1310 * 1311 * 1312 * 1313 * 1314
```

* 71 * QUANTITIES NECESSARY FOR THE PATH ALGORITHM. THESE ARE WORTH MEASURE,
* 72 * PREDECESSOR NODE NUMBER, CUMULATIVE TIME, CUMULATIVE WORTH MEASURE,
* 73 * TIME MEASURE, SOLUTION-VECTOR POINTER, LANE USED, RISK MEASURE, AND
* 74 * PARK TIME. THESE VALUES ARE HELD IN THE NODE-VECTOR FOR EASY REFER-
* 75 * ENCE.
* 76 *
* 77 * If TIME CONFLICT RESOLUTION IS AUTOMATIC
* 78 * REINITIALIZE "WORKING LIST" NODE-VECTOR ENTRIES
* 79 * SET K TO FIRST NODE TO BE CONSIDERED (DEPENDENT ON MOVEMENT TYPE)
* 80 * Else TIME CONFLICT RESOLUTION IS NOT AUTOMATIC
* 81 * If THIS UNIT'S MOVEMENT TYPE, ENTERED TIME, AND UNIT TYPE DIFFERS FROM/
* 82 * THE PREVIOUS UNIT'S
* 83 * REINITIALIZE "WORKING LIST" NODE-VECTOR ENTRIES
* 84 * SET K TO FIRST NODE TO BE CONSIDERED (DEPENDENT ON MOVEMENT TYPE)
* 85 * Endif THIS UNIT'S PARAMETERS DIFFER
* 86 * Endif TIME CONFLICT RESOLUTION IS AUTOMATIC
* 87 *
* 88 * During this algorithm, K is the node from which the algorithm starts.
* 89 * This is determined by the movement type to be either the start node
* 90 * (type is forward) or the destination node (type is backward). I is the
* 91 * final node, the opposite of K (if K is the start node, I is the desti-
* 92 * nation, and vice versa). J is the node under current consideration by
* 93 * the algorithm.
* 94 *
* 95 * Set I to end node (dependent upon movement type)
* 96 * Set J to K
* 97 * Do until node I is labeled ... test immediately for early exit
* 98 * If I is labeled
* 99 * Until node I is labeled
* 00 * Endif I is labeled
* 01 * If J isn't labeled
* 02 * Label node J
* 03 * Set J's cumulative time
* 04 * Do for each of J's adjacent nodes
* 05 * If it is labeled
* 06 * Cycle for each of J's adjacent nodes
* 07 * Else it is not labeled
* 08 * Calculate and assign time and worth values

TRI, INC.
21 FEB 78

LOCATION AND MOVEMENT ANALYSIS SYSTEM
LAMAS FLOW

PAGE 53.003

```
* 09      endif IT IS LABELED
* 10      enddo FOR EACH OF J'S ADJACENT NODES
* 11      endif J ISN'T LABELED
57 * 12      COMPUTE NEXT NODE TO LABEL ...DEPENDENT UPON TYPE
* 13      SET J TO NEW NODE
* 14      enddo UNTIL NODE I LABELED
* 15      COMPUTE SOLUTION-VECTORS AND ROUTE-VECTOR
* 16      enddo FOR EACH UNIT
* 17      return TO CALLING PROGRAM
* 18      enddo UPON FUNCTION REQUEST
*
* 1353
* 1354
* 1355
* 1356
* 1357
* 1358
* 1359
* 1360
* 1361
* 1362
```

***** TOO MANY LINES IN SEGMENT *****

PURGE ALL ROUTES

REF PAGE

```
* 1 * THIS ROUTINE WILL GET RID OF ALL PREVIOUS PATHS WHICH HAVE BEEN
* 2 * CALCULATED. THIS CONSISTS OF ERASING ALL SOLUTION+VECTORS AND
* 3 * .ROUTE+VECTORS, RESETTING THEIR COUNTERS TO ZERO, AND ERASING ALL POINT-
* 4 * .ERS WHICH REFERENCE THESE ARRAYS.
* 5 *
* 6 do FOR EACH ROUTE+VECTOR
* 7   SET CONTENTS TO 0
* 8   enddo FOR EACH ROUTE+VECTOR
* 9   do FOR EACH SOLUTION+VECTOR
*10    SET CONTENTS TO 0
*11   enddo FOR EACH SOLUTION+VECTOR
*12   SET COUNTERS TO 0
*13   dc FOR EACH NODE+VECTOR
*14    SET SOLUTION+VECTOR POINTER TO 0
*15   enddo FOR EACH NODE+VECTOR
```

```
* 1364
* 1365
* 1366
* 1367
* 1368
* 1369
* 1370
* 1371
* 1372
* 1373
* 1374
* 1375
* 1376
* 1377
* 1378
```

TRI, INC. 21 FEB 78
LOCATION AND MOVEMENT ANALYSIS SYSTEM
LAMAS FLDN

PAGE 55

REINITIALIZE 'WORKING LIST' NODE+VECTOR ENTRIES

CALCULATE AND ASSIGN TIME AND MERTH VALUES

```

* 1 WHEN A NODE IS FOUND BY THE PATH ALGORITHM TO BE THE NEXT TO BE
* 2 LABELED, THE TIME AND WORTH VALUES OF ALL OF ITS LINKS TO UNLABELED
* 3 ADJACENT NODES ARE CALCULATED AND ASSIGNED. THAT PROCESS OF CALCULAT-
* 4 ING AND ASSIGNING IS DONE BY THIS ROUTINE.
* 5 THERE ARE TWO TYPES OF TIME AND WORTH: MEASURE AND CUMULATIVE. TIME
* 6 MEASURE AND WORTH MEASURE REFER TO THE VALUE CALCULATED FOR THE LINK
* 7 BETWEEN A NODE AND ONE OF ITS UNLABELED ADJACENT NODES. CUMULATIVE
* 8 TIME AND CUMULATIVE WORTH REFER TO THE ACCUMULATED TIME AND WORTH
* 9 WHICH HAS BUILT UP DURING THE PATH CALCULATION. THUS CUMULATIVE TIME
* 10 IS THE TIME OF DAY AT WHICH A UNIT ARRIVES AT THE ADJACENT NODE, AND
* 11 CUMULATIVE WORTH IS THE WORTH OF THE PATH AT THAT POINT IN TIME.
* 12 WORTH IS A COMBINATION OF RISK AND TIME, USING PARAMETERS SPECIFIED
* 13 BY THE USER. THE EQUATION IS  $C * RISK + K * TIME$ , WHERE C MAY BE
* 14 0 OR 1, AND K MAY BE BETWEEN 0 AND 9.99, BUT BOTH MAY NOT EQUAL 0
* 15 AT THE SAME TIME. A BEST PATH IS A PATH WHICH MINIMIZES WORTH. BY
* 16 ADJUSTING THE PARAMETERS C AND K, THE USER MAKES HIS OWN DEFINITION
* 17 OF BEST. IT COULD BE THAT BEST IS THE LEAST RISKY ( $C=1, K=0$ ), FARTHEST
* 18 ( $C=0, K=1$ ), OR A COMBINATION OF THE TWO.
* 19 RISK IS CALCULATED BY SUMMING THE NUMERICAL VALUES ASSIGNED TO THE
* 20 FOUR RISK FACTORS: GROUND COVER, TERRAIN, BRIDGE, AND CITY CODES.
* 21
* 22 CALCULATE LINK TRAVERSAL TIME
* 23 IF MOVEMENT TYPE IS FORWARD
* 24 CUMULATIVE TIME IS PREDECESSOR'S CUMULATIVE TIME + LINK TIME
* 25 else MOVEMENT TYPE IS BACKWARD
* 26 CUMULATIVE TIME IS THIS NODE'S CUMULATIVE TIME - LINK TIME
* 27 endif MOVEMENT TIME IS FORWARD
* 28
* 29 .RESOLVE TIME CONFLICTS
* 30
* 31 IF USER DOES NOT WANT TIME CONFLICTS RESOLVED
* 32 CONTINUE

```

```
* 33 elseif THIS NODE+VECTOR HAS NEVER BEEN USED BY A PATH
* 34   CONTINUE
* 35 else THIS NODE+VECTOR HAS BEEN USED BY A PATH
* 36
* 37   LOOK FOR TIME CONFLICT. IF ONE IS FOUND, RESOLVE IT
* 38
* 39   SET ALL FOUR LANES TO CUMMULATIVE TIME
* 40   do FOR EACH SOLUTION+VECTOR REFERING TO THIS NODE
* 41     if THE TIME INTERVAL SHOULD NOT BE CONSIDERED
* 42       cycle FOR EACH SOLUTION+VECTOR
* 43     else THE TIME INTERVAL SHOULD BE CONSIDERED
* 44       SET INDEX TO SOLUTION+VECTOR 'S LANE NUMBER
* 45       if THIS UNIT CAN CLEAR THIS NODE BEFORE THE TIME INTERVAL
* 46         cycle FOR EACH SOLUTION+VECTOR
* 47         if THIS UNIT CANNOT CLEAR THIS NODE BEFORE THE TIME INTERVAL
* 48           if THIS UNIT DOES NOT ARRIVE UNTIL AFTER THE TIME INTERVAL
* 49             cycle FOR EACH SOLUTION+VECTOR
* 50           else THIS UNIT ARRIVES DURING THE TIME INTERVAL
* 51
* 52   THERE IS A TIME CONFLICT
* 53
* 54   if MOVEMENT TYPE IS FORWARD
* 55     HOLD UP UNIT UNTIL NODE CLEARS
* 56     RECORD THIS TIME IN LANE (INDEX)
* 57   else MOVEMENT TYPE IS BACKWARD
* 58     HAVE UNIT ARRIVE BEFORE TIME INTERVAL
* 59     RECORD THIS TIME IN LANE (INDEX)
* 60   endif MOVEMENT TYPE IS FORWARD
* 61
* 62   THAT'S ALL FOR THIS SOLUTION+VECTOR
* 63
* 64   endif THIS UNIT DOES NOT ARRIVE UNTIL AFTER THE TIME INTERVAL
* 65   endif THIS UNIT CAN CLEAR THIS NODE BEFORE THE TIME INTERVAL
* 66   endif THE TIME INTERVAL SHOULD NOT BE CONSIDERED
* 67   enddo FOR EACH SOLUTION+VECTOR REFERING TO THIS NODE
* 68
* 69   FIND THE BEST LANE (FOR FORWARD, BEST IS LEAST TIME, FOR BACKWARD, BEST
* 70   IS THE GREATEST TIME).
```

```
* 71      if MOVEMENT IS FORWARD          1465
* 72          SET BEST TO 32767          1466
* 73      else MOVEMENT IS BACKWARD      1467
* 74          SET BEST TO 0             1468
* 75      endif MOVEMENT IS FORWARD     1469
* 76          do FOR EACH AVAILABLE LANE FOR THIS LINK   1470
* 77              if TIME OF LANE IS LESS (GREATER) THAN BEST    1471
* 78                  SET BEST TO TIME OF LANE      1472
* 79          endif TIME OF LANE IS LESS (GREATER) THAN BEST    1473
* 80          REMEMBER LANE NUMBER          1474
* 81      enddo FOR EACH AVAILABLE LANE FOR THIS LINK    1475
* 82
* 83      ANY TIME CONFLICT HAS NOW BEEN RESOLVED. THE BEST RESULT FROM THE 1476
* 84      .PREVIOUS SECTION IS NOW THE 'TIME OF RECORD'.          1477
* 85
* 86      CALCULATE PARKING TIME (*TIME OF RECORD* LESS CUMULATIVE TIME) 1478
* 87
* 88      .PARKING TIME AFFECTS THE ENTIRE LENGTH OF THE UNIT, SO IT MUST BE 1479
* 89      .SEEN IF THE ADDITION OF THIS PARKING TIME CREATES ANY NEW CONFLICTS. 1480
* 90      .IF SO, THIS LINK CANNOT BE USED.          1481
* 91
* 92      if PARKING TIME EQUALS ZERO          1482
* 93          CONTINUE          1483
* 94      else PARKING TIME DOES NOT EQUAL ZERO          1484
* 95          if PARKING TIME CREATES NEW CONFLICTS      1485
* 96              MAKE NO ASSIGNMENTS IF TIME AND WORTH 1486
* 97              return          1487
* 98          endif PARKING TIME CREATES NEW CONFLICTS 1488
* 99          endif PARKING TIME EQUALS ZERO          1489
* 100         endif THE USER DOES NOT WANT TIME CONFLICTS RESOLVED 1490
* 101
* 102         NOW CALCULATE RISK AND MAKE ASSIGNMENTS          1491
* 103
* 104         CALCULATE RISK          1492
* 105         CALCULATE WORTH FOR THIS LINK          1493
* 106         CALCULATE CUMULATIVE WORTH          1494
* 107         if CUMULATIVE WORTH IS LESS THAN CURRENT CUMULATIVE WORTH 1495
* 108
```

TRW, INC.
21 FEB 78

LOCATION AND MOVEMENT ANALYSIS SYSTEM
1 MARS FLINN

PAGE 56.003

```
* 39      MAKE ASSIGNMENTS OF TIME, WORTH, PARKING, AND LANE
* 10      endif CUMULATIVE WORTH IS LESS THAN CURRENT CUMULATIVE WORTH
*
```

```
***** TOO MANY LINES IN SEGMENT
```

1503
1504

COMPUTE NEXT NODE TO LABEL

REF PAGE

```
* 1      THIS CONSISTS OF FINDING WHICH UNLABELED NODE VECTOR HAS THE
* 2      LEAST CUMULATIVE WORTH. BY MAKING WORTH A FUNCTION SUCH THAT
* 3      THE BETTER A NODE'S WORTH, THE SMALLER THE WORTH VALUE, CUM-
* 4      .WORTH MUST BE ORIGINALLY SET TO INFINITY, OR IN THIS
* 5      CASE, 32767.
* 6
* 7
* 8      SET LEAST TO 32767
* 9      do FOR EACH NODE
* 10     if NODE IS LABELED
* 11     cycle
* 12     elseif WORTH MEASURE < LEAST
* 13     SET LEAST TO WORTH MEASURE
* 14     SAVE NODE NUMBER
* 15     endif NODE IS LABELED
* 16     enddo FOR EACH NODE
*
* 1506
* 1507
* 1508
* 1509
* 1510
* 1511
* 1512
* 1513
* 1514
* 1515
* 1516
* 1517
* 1518
* 1519
* 1520
* 1521
```

EXPLANATION OF PATH DETERMINATION ALGORITHM

1 OUR PROBLEM IS TO DETERMINE A ROUTE FOR A GROUND FORCE UNIT TO # 1523
2 FOLLOW, SUCH THAT THE PATH GETS THE UNIT TO ITS DESTINATION NODE FROM # 1524
3 ITS STARTING NODE AS QUICKLY AND/OR AS LEAST RISKY AS POSSIBLE. THIS # 1525
4 IS A SHORTEST PATH PROBLEM, WITH SOME TWISTS THROWN IN. ONE OF THESE # 1526
5 TWISTS IS THAT EACH UNIT HAS A DIFFERENT RATE OF TRAVEL, WITH THAT RATE # 1527
6 VARYING ACCORDING TO THE TIME OF DAY. ANOTHER IS THAT WE REALLY HAVE # 1528
7 MORE THAN ONE UNIT, AND WHEN MORE THAN ONE UNIT HAS THE SAME DESTINATION # 1529
8 NODE, WE MAY WISH FOR THE UNITS TO ARRIVE THERE AT THE SAME TIME. WE HAVE # 1530
9 FOUND THAT THE SIMULTANEOUS DESTINATION TIME PROBLEM MAY BE SOLVED BY STARTING # 1531
10 AT THE DESTINATION NODE AND WORKING OUR WAY BACK TO THE STARTING NODE, RUN- # 1532
11 NING TIME IN REVERSE. # 1533
12 PERHAPS IT WOULD BE BEST TO FIRST EXPLAIN DIJKSTRA'S ALGORITHM FOR ONE UNIT # 1534
13 SUPPOSE THAT THERE ARE N NODES, NUMBERED FROM 1 TO N. AND SUPPOSE FURTHER # 1535
14 THAT WE WISH TO FIND THE SHORTEST PATH BETWEEN NODES 1 AND N. THAT IS, NODE 1 # 1536
15 IS THE DESTINATION, AND NODE N IS THE STARTING NODE. LABEL 1 WITH THE PERM- # 1537
16 ANENT VALUE ZERO, AND TENTATIVELY LABEL ALL OTHERS WITH VALUE INFINITY. EACH # 1538
17 TIME A PERMANENT LABEL IS ASSIGNED, IT REPRESENTS THE SHORTEST DISTANCE BETWEEN # 1539
18 THAT NODE AND THE DESTINATION NODE, NODE N. # 1540
19 ONE BY ONE, COMPARE EACH NODE LABEL EXCEPT THAT AT 1 WITH THE SUM OF THE # 1541
20 LABEL OF NODE 1 (THAT IS, 0) AND THE DIRECT DISTANCE FROM NODE 1 TO THE NODE # 1542
21 IN QUESTION. THE SMALLER OF THE TWO NUMBERS IS THE NEW TENTATIVE LABEL. # 1543
22 NEXT, DETERMINE THE SMALLEST OF THE N-1 TENTATIVE LABELS AND DECLARE IT # 1544
23 PERMANENT. SUPPOSE THAT NODE K IS THE ONE PERMANENTLY LABELED. THEN, # 1545
24 ONE AT A TIME, COMPARE EACH OF THE N-2 REMAINING TENTATIVE NODE LABELS TO # 1546
25 THE SUM OF THE LABEL JUST ASSIGNED PERMANENTLY TO NODE K AND THE DIRECT # 1547
26 DISTANCE FROM NODE K TO THE NODE UNDER CONSIDERATION. THE SMALLER OF THE TWO # 1548
27 NUMBERS BECOMES THE TENTATIVE LABEL. DETERMINE THE MINIMUM OF THE N-2 TEN- # 1549
28 TATIVE LABELS, DECLARE IT PERMANENT, AND MAKE IT THE BASIS OF ANOTHER MOD- # 1550
29 IFICATION OF THE REMAINING TENTATIVE LABELS OF THE TYPE DESCRIBED ABOVE. # 1551
30 WHEN, AFTER AT MOST N-1 EXECUTIONS OF THE FUNDAMENTAL ITERATIVE STEP, NODE # 1552
31 N IS PERMANENTLY LABELED, THE PROCEDURE TERMINATES. # 1553
32 THE PATH FROM N TO 1 MAY BE RECONSTRUCTED IF, FOR EACH LABELED NODE, THE # 1554
33 NODE FROM WHICH IT WAS LABELED (ITS PREDECESSOR) IS RECORDED. THEN, BY START- # 1555
34ING AT N, WE MAY SIMPLY FOLLOW THE PREDECESSORS BACK TO THE DESTINATION NODE. # 1556
35 THIS PROCESS IS CORRECT, BUT FOR EASE OF HANDLING ON THE COMPUTER, CERTAIN # 1557
36 ADJUSTMENTS HAVE BEEN MADE. THESE WERE ORIGINALLY SUGGESTED BY MINTY AND OTHERS # 1558

37 INSTEAD OF CONSIDERING A LABEL TO POSSESS A VALUE, WE SAY THAT A NODE IS EITHER
38 LABELED OR IT IS NOT, AND WE CONSIDER THE DISTANCE FROM NODE 1 TO BE A SEPARATE
39 ENTITY. AND INSTEAD OF USING INFINITY, WE USE THE LARGEST NUMBER ALLOWABLE.
40 WE ALSO DO SOMETHING WHICH WASN'T ANTICIPATED BY THESE GENTLEMEN, AND THAT
41 IS TO SOMETIMES RUN THE ALGORITHM IN REVERSE. NOW THE BASIC STEPS ARE STILL
42 THE SAME EXCEPT THAT WE ARE RUNNING TIME IN REVERSE, USING AN ARRIVAL TIME
43 AS THE STARTING POINT. THUS, ALL DISTANCES ARE NEGATIVE, AND INSTEAD OF
44 LOOKING FOR THE LEAST "LABEL", WE LOOK FOR THE GREATEST. BY DOING THIS,
45 WE FIND OUT AT WHAT TIME A UNIT HAD TO LEAVE ITS STARTING NODE IN ORDER TO
46 ARRIVE AT ITS DESTINATION NODE AT THE GIVEN TIME.
47 THUS WE HAVE TWO CHOICES IN APPROACH TO THIS PROBLEM. WE MAY WISH TO FIND
48 THE SHORTEST PATH FROM NODE 1 TO NODE N, OR WE MAY FIND WHICH NODE IS SO FAR
49 FROM NODE N.
50 RATHER THAN WORK WITH STRICTLY DISTANCE, WE WILL REALLY BE WORKING WITH
51 TIME, AND THE TWO QUESTIONS WE WILL ADDRESS WILL BE 1) HOW FAST CAN WE GET FROM
52 NODE 1 TO NODE N? AND 2) GIVEN A TIME WE WISH TO ARRIVE AT NODE N, AT WHAT TIME
53 SHOULD WE LEAVE NODE 1?
#

1559
4560
4561
4562
4563
4564
4565
4566
4567
4568
4569
4570
4571
4572
4573
4574
4575
#

#####

COMPUTE SOLUTION-VECTORS AND ROUTE-VECTOR

REF PAGE

```
*      1   * SOLUTION-VECTORS ARE CONSTRUCTED BY TRACING THE PATH OF NODE-VECTORS.  
*      2   * THIS IS DONE BY STARTING AT NODE 1 AND FOLLOWING ITS PREDECESSOR POINT-  
*      3   * ER, THEN THE PREDECESSOR'S PREDECESSOR POINTER, ETC., UNTIL NODE K IS  
*      4   * REACHED. EACH TIME A NEW PREDECESSOR IS REACHED, A NEW SOLUTION-VECTOR  
*      5   * IS CREATED. ITS APPROPRIATE VALUES ARE ENTERED, AND IT IS LINKED TO  
*      6   * ANY OTHER SOLUTION-VECTORS THAT REFER TO THE SAME NODE. AT THE SAME  
*      7   * TIME AS THE SOLUTION-VECTORS ARE BEING CONSTRUCTED, THE PATH'S ROUTE-  
*      8   * VECTOR IS ALSO BEING MADE. THIS IS BECAUSE MANY OF ITS ENTRIES ARE  
*      9   * ACCUMULATIONS OF INFORMATION GAINED AT EACH NODE.  
*     10   * ONE WORD OF NOTE, FOR UNIFORMITY, THE SOLUTION-VECTORS OF ALL FORWARD-  
*     11   * WORD PATHS REMAIN AS THEY ARE CONSTRUCTED, BUT THE BACKWARD PATH SOLUTION-  
*     12   * VECTORS HAVE THEIR ORDER REVERSED, THUS, THERE IS NO DATA BASE  
*     13   * DISTINCTION BETWEEN BACKWARD AND FORWARD PATHS.  
*     14   *  
*     15   * INCREMENT ROUTE NUMBER COUNTER  
*     16   * SET THE UNIT NUMBER  
*     17   * SET TOTAL ROUTE TIME TO 0  
*     18   * SET TOTAL RISK MEASURE TO 0  
*     19   * SET NUMBER OF NODES TO 1  
*     20   * SET ROUTE MOVEMENT TYPE  
*     21   * SET THE STARTING TIME  
*     22   * SET J TO 1  
do UNTIL NODE K IS REACHED  
*     23   *  
*     24   * INCREMENT SOLUTION-VECTOR COUNTER  
*     25   * SET NODE NUMBER OF SOLUTION-VECTOR TO J  
*     26   * SET LANE NUMBER AND PARK TIME  
*     27   * SET CODE FOR CONFLICT RESOLUTION CONSIDERATION  
*     28   * INCREMENT ROUTE DISTANCE BY LINK DISTANCE FROM J TO PREDECESSOR NODE  
*     29   * INCREMENT TOTAL ROUTE TIME BY LINK TIME VALUE  
*     30   * INCREMENT TOTAL RISK BY LINK RISK VALUE  
*     31   * SET TIME IN ACCORDING TO ROUTE TYPE (FORWARD OR BACKWARD)  
*     32   * SET TIME INT ACCORDING TO ROUTE TYPE
```

```
* 33      INCREMENT NUMBER OF SOLUTION+VECTORS          1609
* 34      SET J TO ITS PREDECESSOR          1610
* 35      IF J IS NOW K          1611
* 36          UNDO UNTIL NODE K IS REACHED          1612
* 37      ENDIF J IS NOW K          1613
* 38          UNDO UNTIL NODE K IS REACHED          1614
* 39
* 40      .REVERSE THE ORDER OF THE SOLUTION+VECTORS IF NECESSARY          1615
* 41
* 42      IF MOVEMENT TYPE IS BACKWARD          1616
* 43          REVERSE ORDER OF SOLUTION+VECTORS          1617
* 44      ENDIF MOVEMENT TYPE IS BACKWARD          1618
* 45
* 46      IF SOLUTION+VECTOR POINTER OF NODE+VECTOR = 0          1619
* 47          SET SOLUTION+VECTOR POINTER TO SOLUTION+VECTOR COUNTER          1620
* 48          SET MULTIPLE USE OF SOLUTION+VECTOR TO -1          1621
* 49      ELSE POINTER DOESN'T EQUAL 0          1622
* 50          DO FOREVER          1623
* 51              FOLLOW POINTER          1624
* 52                  IF MULTIPLE USE OF POINTER = -1          1625
* 53                      SET MULTIPLE USE OF POINTER TO SOLUTION+VECTOR COUNTER          1626
* 54                      SET MULTIPLE USE OF SOLUTION+VECTOR TO -1          1627
* 55                  UNDO FOREVER          1628
* 56                  ENDIF MULTIPLE USE OF POINTER = -1          1629
* 57          ENDIF FOREVER          1630
* 58      ENDIF SOLUTION+VECTOR POINTER OF NODE+VECTOR = 0          1631
* 59      SET ROUTE LIST HEAD POINTER          1632
* 60      IF MOVEMENT TYPE IS FORWARD          1633
* 61          ADD PARKING TIMES WHERE APPROPRIATE          1634
* 62      ENDIF MOVEMENT TYPE IS FORWARD          1635
* 63          PROMPT FOR ENTRY OF PRINTING OPTION          1636
* 64          IF USER WANTS PRINTING          1637
* 65              PRINT CONTENTS OF SOLUTION+VECTORS          1638
* 66              PRINT ANY TIME CONFLICTS          1639
* 67          ENDIF USER WANTS PRINTING          1640
* 68          RETURN TO CALLING PROGRAM          1641
* 69
```

CALCULATE SECOND BEST PATH

REF PAGE

```
*      1   THIS IS DONE BY SYSTEMATICALLY "ERASING" NODES ALONG THE BEST PATH          * 1646
*      2   .AND CALCULATING THE NEW BEST PATH. AFTER THE NEW PATH IS CALCULATED,      * 1647
*      3   .THE ERASED NODE IS RESTORED, AND THE NEXT ERASED, AND SO ON. THE NEW      * 1648
*      4   .PATH WITH THE COMPLETION TIME CLOSEST TO THE INITIAL BEST PATH IS THEN      * 1649
*      5   .THE SECOND BEST PATH. FOR ADDED INSIGHT, WE WILL KEEP THE NEXT THREE      * 1650
*      6   .BEST PATHS, HOWEVER, IT IS OFTEN THE CASE THAT TWO (OR EVEN ALL THREE)      * 1651
*      7   .OF THE NEXT BEST PATHS ARE IDENTICAL.                                * 1652
*      8
*      9   do UPON FUNCTION REQUEST                                         * 1653
*      10  PURGE ALL ROUTES                                         * 1654
*      11
*      12  .THIS ROUTINE CREATES THREE FICTIONAL UNITS TO BE ASSOCIATED WITH THE      * 1655
*      13  .THREE NEW CALCULATED ROUTES. CHECK FOR STORAGE ROOM IN THE UNITS DATA      * 1656
*      14  .BASE.                                         * 1657
*      15
*      16  .if THERE IS NOT ENOUGH ROOM IN UNITS FOR THREE MORE UNIT-VECTORS      * 1658
*      17  .PRINT ERROR MESSAGE                                         * 1659
*      18  .return TO CALLING PROGRAM                                         * 1660
*      19  .endif THERE IS NOT ENOUGH ROOM IN UNITS FOR THREE MORE UNIT-VECTORS      * 1661
*      20  .PROMPT FOR UNIT NAME                                         * 1662
*      21  .if INPUT IS NULL.                                         * 1663
*      22  .return TO CALLING PROGRAM                                         * 1664
*      23  .endif INPUT IS NULL.                                         * 1665
*      24  .if ENTERED UNIT NAME IS NOT IN THE DATA BASE      * 1666
*      25  .PRINT ERROR MESSAGE                                         * 1667
*      26  .TRY AGAIN                                         * 1668
*      27  .else ENTERED UNIT NAME IS LISTED
*      28  .PROMPT FOR STARTING TIME                                         * 1669
*      29  .PROMPT FOR RISK AND TIME PARAMETERS ...C AND K      * 1670
*      30  .if DESTINATION IS FIBA AND NONE HAS BEEN DEFINED      * 1671
*      31  .PRINT ERROR MESSAGE                                         * 1672
*      32
*      33
*      34
*      35
*      36
*      37
*      38
*      39
*      40
*      41
*      42
*      43
*      44
*      45
*      46
*      47
*      48
*      49
*      50
*      51
*      52
*      53
*      54
```

```
* 33 * * * * 1678
* 34 * * * * 1679
* 35 * * * * 1680
* 36 * * * * 1681
* 37 * * * * 1682
* 38 * * * * 1683
* 39 * * * * 1684
do FOR SECOND THROUGH NEXT-TO-LAST NODES OF THE BEST PATH 1685
* 40 * * * * 1685
ERASE ROADS LEADING TO THE NODE
* 41 * * * * 1686
REINITIALIZE "WORKING LIST" NODE+VECTOR ENTRIES
* 42 * * * * 1687
CALCULATE BEST PATH ...FORWARD WITH NO DECONFLICTING
COMPARE COMPLETION TIME WITH OTHERS IN TIME ARRAY
* 43 * * * * 1688
IF THIS TIME IS LESS THAN ANY ONE TIME IN ARRAY
* 44 * * * * 1689
PLACE THIS TIME IN ARRAY
* 45 * * * * 1690
ORDER ARRAY IN ASCENDING FASHION
* 46 * * * * 1691
REMEMBER ERASED NODE
* 47 * * * * 1692
endif THIS TIME IS LESS THAN ANY ONE TIME IN ARRAY
* 48 * * * * 1693
RESTORE ERASED NODE'S ROADS
* 49 * * * * 1694
enddo FOR SECOND THROUGH NEXT-TO-LAST NODES OF THE BEST PATH
* 50 * * * * 1695
* 51 * * * * 1696
* 52 * * * * 1697
* 53 * * * * 1698
* 54 * * * * 1699
* 55 * * * * 1700
do FOR EACH ENTRY IN TIME ARRAY
* 56 * * * * 1701
ERASE REMEMBERED NODE
* 57 * * * * 1702
REINITIALIZE "WORKING LIST" NODE+VECTOR ENTRIES
* 58 * * * * 1703
CALCULATE BEST PATH ...FORWARD WITH NO DECONFLICTING
* 59 * * * * 1704
COMPUTE SOLUTION+VECTORS AND ROUTE+VECTOR
* 60 * * * * 1705
CREATE NEW UNIT+VECTOR
* 61 * * * * 1706
ASSOCIATE NEW UNIT+VECTOR WITH THIS ROUTE
* 62 * * * * 1707
PRINT MAP+NUMBER, NODE+NUMBER OF ERASED NODE
* 63 * * * * 1708
PRINT UTM LATITUDE, LONGITUDE OF ERASED NODE
* 64 * * * * 1709
PRINT DELAY TIME
* 65 * * * * 1710
RESTORE ERASED NODE
* 66 * * * * 1711
enddo FOR EACH ENTRY IN TIME ARRAY
* 67 * * * * 1712
endif ENTERED UNIT NAME IS NOT IN THE DATA BASE
* 68 * * * * 1713
enddo UPON FUNCTION REQUEST
```

FIND THE ORDER OF PRIORITIES FOR BEST NETWORK TRAVEL

REF PAGE

```
* 1   THIS PROBLEM IS: GIVEN N UNITS TRAVELING THROUGH A ROAD NETWORK, AND      * 1715
* 2   GIVEN THAT WHEN ANY TWO UNITS WISH TO USE THE SAME NODE DURING OVER-      * 1716
* 3   LAPPING TIME INTERVALS THE ONE WITH THE HIGHER PRIORITY USES IT FIRST      * 1717
* 4   WHILE THE OTHER WAITS OR GIVES AROUND, WHAT ORDERING OF PRIORITIES PRO-      * 1718
* 5   DUCES THE BEST RESULTS?      * 1719
* 6   THIS ROUTINE ANSWERS THIS QUESTION BY TRYING EVERY POSSIBLE COMBINA-      * 1720
* 7   TION OF PRIORITY ORDERINGS, AND SEEING WHICH ONE DOES IN FACT PRODUCE      * 1721
* 8   THE BEST RESULTS. BY "BEST" WE MEAN THAT THE EARLIEST START TIME OF      * 1722
* 9   THE UNITS IN THE CHOSEN PRIORITY SCHEME IS LATER THAN ALL OTHER EAR-      * 1723
* 10  LIEST TIMES PRODUCED USING THE OTHER PRIORITY SCHEMES.      * 1724
* 11
* 12  do UPON FUNCTION REQUEST      * 1725
* 13  PURGE ALL ROUTE'S      * 1726
* 14  PROMPT FOR TIME MOVEMENT ... FORWARD GIVES RATHER UNINTERESTING RESULTS      * 1727
* 15  if MOVEMENT IS FORWARD      * 1728
* 16    PROMPT FOR STARTING TIME      * 1729
* 17  else MOVEMENT IS BACKWARD      * 1730
* 18    if DESTINATION IS THE FEBA AND A FEBA HAS NOT BEEN DEFINED      * 1731
* 19      PRINT ERROR MESSAGE      * 1732
* 20      TRY AGAIN      * 1733
* 21      endif DESTINATIN IS THE FEBA      * 1734
* 22      PROMPT FOR ARRIVAL TIME      * 1735
* 23      endif MOVEMENT IS FORWARD      * 1736
* 24
* 25      .OBTAIN UNIT NAMES. AT THIS TIME, THIS ROUTINE WILL HANDLE AT MOST FIVE      * 1737
* 26      .UNITS. THIS IS BECAUSE OF TIME CONSIDERATIONS.      * 1738
* 27
* 28      do AT MOST FIVE TIMES      * 1739
* 29        PROMPT FOR UNIT NAME      * 1740
* 30        if INPUT IS NULL AND NO UNIT NAME HAS BEEN ENTERED      * 1741
* 31          return TO CALLING PROGRAM      * 1742
* 32        endif INPUT IS NULL AND NO UNIT NAME HAS BEEN ENTERED      * 1743
* 33
* 34
* 35
* 36
* 37
* 38
* 39
* 40
* 41
* 42
* 43
* 44
* 45
* 46
* 47
* 48
* 49
* 50
* 51
* 52
* 53
* 54
```

```
* 33      if INPUT IS NULL AND AT LEAST ONE UNIT NAME HAS BEEN ENTERED
* 34          undo AT MOST FIVE TIMES
* 35      else INPUT IS NOT NULL
* 36          if ENTERED UNIT NAME IS NOT IN THE DATA BASE
* 37              PRINT MESSAGE
* 38                  TRY AGAIN
* 39          endif ENTERED UNIT NAME IS NOT IN THE DATA BASE
* 40              REMEMBER UNIT NAME
* 41          endif INPUT IS NULL AND AT LEAST ONE UNIT NAME HAS BEEN ENTERED
* 42      enddo AT MOST FIVE TIMES
* 43
* 44          .GENERATE ALL POSSIBLE PRIORITY COMBINATIONS AND MAKE PATH CALCULATIONS
* 45          .USING TIME.
* 46
* 47          INITIALIZE EARLIEST TIME
* 48          dc FOR EACH COMBINATION OF PRIORITIES
* 49          RESET MOVE+VECTOR SOLUTION+VECTOR POINTERS
* 50          RESET THIS COMBINATION'S EARLIEST START TIME
* 51          do FOR EACH UNIT
* 52              CALCULATE BEST PATH ...WITH DECONFLICTING
* 53              MAKE SOLUTION+VECTORS AND ROUTE+VECTOR
* 54              if THE START TIME IS EARLIER THAN THE OTHERS
* 55                  REMEMBER THE TIME
* 56          endif THIS PATH WORSE THAN OTHERS
* 57          enddo FOR EACH UNIT
* 58          if EARLIEST START TIME IS LATER THAN ALL OTHER EARLIEST TIMES
* 59              REMEMBER PRIORITY COMBINATION
* 60              REMEMBER EARLIEST START TIME
* 61          endif EARLIEST START TIME IS LATER THAN ALL OTHER EARLIEST TIMES
* 62          enddo FOR EACH COMBINATION OF PRIORITIES
* 63
* 64          .CALCULATE RESULTS
* 65
* 66          REINITIALIZE SOLUTION+VECTORS AND ROUTE+VECTORS
* 67          do FOR EACH UNIT USING THE BEST PRIORITY SCHEME
* 68              CALCULATE BEST PATH
* 69              COMPUTE SOLUTION+VECTORS AND ROUTE+VECTOR
* 70          enddo FOR EACH UNIT USING THE BEST PRIORITY SCHEME
* 1747
* 1748
* 1749
* 1750
* 1751
* 1752
* 1753
* 1754
* 1755
* 1756
* 1757
* 1758
* 1759
* 1760
* 1761
* 1762
* 1763
* 1764
* 1765
* 1766
* 1767
* 1768
* 1769
* 1770
* 1771
* 1772
* 1773
* 1774
* 1775
* 1776
* 1777
* 1778
* 1779
* 1780
* 1781
* 1782
* 1783
* 1784
```

TRW, INC.
21 FEB 78

LOCATION AND MOVEMENT ANALYSIS SYSTEM
LAMAS FLOW

PAGE: 61.002

* 71 PRINT MESSAGE INFORMING USER THAT RESULTS ARE AVAILABLE
* 72 enddo UPDN FUNCTION REQUEST
*

* 1785
* 1786
*

CALCULATE BEST NODE AT WHICH TO INTERDICT

REF PAGE

```
* 1      THIS ROUTINE CALCULATES THE NODE AT WHICH IT IS BEST TO INTERDICT,
* 2      WHERE 'BEST' MEANS 'HAS THE GREATEST WORTH DIFFERENCE BETWEEN THE BEST
* 3      PATH AND THE PATH WITH INTERDICTION'. IN ADDITION, THE PROGRAM FINDS
* 4      THE TIME WINDOW DURING WHICH THE INTERDICTION SHOULD OCCUR.
* 5      THIS IS DONE IN A MANNER SIMILAR TO THE PROCESS FOR CALCULATING THE
* 6      SECOND BEST PATH. EACH NODE OF THE BEST PATH IS SYSTEMATICALLY ER-
* 7      ASED, BUT THEN THERE IS A DIFFERENCE. THIS ROUTINE FORCES THE UNIT TO
* 8      TRAVEL A DISTANCE DOWN THE BEST PATH, AND THEN LETS IT GO ITS OWN WAY.
* 9      E.G., SUPPOSE THAT THE FIFTH NODE HAS BEEN ERASED. THERE WILL BE FOUR
* 10     PATHS CALCULATED FOR THIS ONE NODE ERASURE. FIRST, THE UNIT WILL BE
* 11     FORCED TO TRAVEL THROUGH NODE ONE, AND THEN IT MAY GO AS IT PLEASES.
* 12     NEXT, IT IS FORCED THROUGH NODE 2; THEN THROUGH NODE 3, AND FINALLY,
* 13     THROUGH NODE FOUR. IN THIS WAY, WE SIMULATE TIME RELATED INTERDICTION.
* 14
* 15    do UPON FUNCTION REQUEST
* 16        PURGE ALL ROUTES
* 17
* 18        *WE WILL ACTUALLY KEEP THE THREE BEST INTERDICTION CASES IN AN ARRAY.
* 19        *AND, WE WILL CREATE THREE FICTITIOUS UNITS.
* 20
* 21        INITIALIZE BEST ARRAY
* 22        if THERE IS NOT ROOM FOR THREE MORE JUNIT+VECTORS
* 23            PRINT MESSAGE
* 24            RETURN TO CALLING PROGRAM
* 25        endif THERE IS NOT ROOM FOR THREE MOR UNIT+VECTORS
* 26        PROMPT FOR UNIT NAME
* 27        if INPUT IS NULL
* 28            RETURN TO CALLING PROGRAM
* 29        else INPUT IS NOT NULL
* 30            if UNIT NAME IS NOT IN DATA BASE
* 31                PRINT MESSAGE
* 32                    TRY AGAIN
* 33
* 34
* 35
* 36
* 37
* 38
* 39
* 40
* 41
* 42
* 43
* 44
* 45
* 46
* 47
* 48
* 49
* 50
* 51
* 52
* 53
* 54
* 55
* 56
* 57
* 58
* 59
* 60
* 61
* 62
* 63
* 64
* 65
* 66
* 67
* 68
* 69
* 70
* 71
* 72
* 73
* 74
* 75
* 76
* 77
* 78
* 79
* 80
* 81
* 82
* 83
* 84
* 85
* 86
* 87
* 88
* 89
* 90
* 91
* 92
* 93
* 94
* 95
* 96
* 97
* 98
* 99
* 100
* 101
* 102
* 103
* 104
* 105
* 106
* 107
* 108
* 109
* 110
* 111
* 112
* 113
* 114
* 115
* 116
* 117
* 118
* 119
* 120
* 121
* 122
* 123
* 124
* 125
* 126
* 127
* 128
* 129
* 130
* 131
* 132
* 133
* 134
* 135
* 136
* 137
* 138
* 139
* 140
* 141
* 142
* 143
* 144
* 145
* 146
* 147
* 148
* 149
* 150
* 151
* 152
* 153
* 154
* 155
* 156
* 157
* 158
* 159
* 160
* 161
* 162
* 163
* 164
* 165
* 166
* 167
* 168
* 169
* 170
* 171
* 172
* 173
* 174
* 175
* 176
* 177
* 178
* 179
* 180
* 181
* 182
* 183
* 184
* 185
* 186
* 187
* 188
* 189
* 190
* 191
* 192
* 193
* 194
* 195
* 196
* 197
* 198
* 199
* 200
* 201
* 202
* 203
* 204
* 205
* 206
* 207
* 208
* 209
* 210
* 211
* 212
* 213
* 214
* 215
* 216
* 217
* 218
* 219
* 220
* 221
* 222
* 223
* 224
* 225
* 226
* 227
* 228
* 229
* 230
* 231
* 232
* 233
* 234
* 235
* 236
* 237
* 238
* 239
* 240
* 241
* 242
* 243
* 244
* 245
* 246
* 247
* 248
* 249
* 250
* 251
* 252
* 253
* 254
* 255
* 256
* 257
* 258
* 259
* 260
* 261
* 262
* 263
* 264
* 265
* 266
* 267
* 268
* 269
* 270
* 271
* 272
* 273
* 274
* 275
* 276
* 277
* 278
* 279
* 280
* 281
* 282
* 283
* 284
* 285
* 286
* 287
* 288
* 289
* 290
* 291
* 292
* 293
* 294
* 295
* 296
* 297
* 298
* 299
* 300
* 301
* 302
* 303
* 304
* 305
* 306
* 307
* 308
* 309
* 310
* 311
* 312
* 313
* 314
* 315
* 316
* 317
* 318
* 319
* 320
* 321
* 322
* 323
* 324
* 325
* 326
* 327
* 328
* 329
* 330
* 331
* 332
* 333
* 334
* 335
* 336
* 337
* 338
* 339
* 340
* 341
* 342
* 343
* 344
* 345
* 346
* 347
* 348
* 349
* 350
* 351
* 352
* 353
* 354
* 355
* 356
* 357
* 358
* 359
* 360
* 361
* 362
* 363
* 364
* 365
* 366
* 367
* 368
* 369
* 370
* 371
* 372
* 373
* 374
* 375
* 376
* 377
* 378
* 379
* 380
* 381
* 382
* 383
* 384
* 385
* 386
* 387
* 388
* 389
* 390
* 391
* 392
* 393
* 394
* 395
* 396
* 397
* 398
* 399
* 400
* 401
* 402
* 403
* 404
* 405
* 406
* 407
* 408
* 409
* 410
* 411
* 412
* 413
* 414
* 415
* 416
* 417
* 418
* 419
* 420
* 421
* 422
* 423
* 424
* 425
* 426
* 427
* 428
* 429
* 430
* 431
* 432
* 433
* 434
* 435
* 436
* 437
* 438
* 439
* 440
* 441
* 442
* 443
* 444
* 445
* 446
* 447
* 448
* 449
* 450
* 451
* 452
* 453
* 454
* 455
* 456
* 457
* 458
* 459
* 460
* 461
* 462
* 463
* 464
* 465
* 466
* 467
* 468
* 469
* 470
* 471
* 472
* 473
* 474
* 475
* 476
* 477
* 478
* 479
* 480
* 481
* 482
* 483
* 484
* 485
* 486
* 487
* 488
* 489
* 490
* 491
* 492
* 493
* 494
* 495
* 496
* 497
* 498
* 499
* 500
* 501
* 502
* 503
* 504
* 505
* 506
* 507
* 508
* 509
* 510
* 511
* 512
* 513
* 514
* 515
* 516
* 517
* 518
* 519
* 520
* 521
* 522
* 523
* 524
* 525
* 526
* 527
* 528
* 529
* 530
* 531
* 532
* 533
* 534
* 535
* 536
* 537
* 538
* 539
* 540
* 541
* 542
* 543
* 544
* 545
* 546
* 547
* 548
* 549
* 550
* 551
* 552
* 553
* 554
* 555
* 556
* 557
* 558
* 559
* 560
* 561
* 562
* 563
* 564
* 565
* 566
* 567
* 568
* 569
* 570
* 571
* 572
* 573
* 574
* 575
* 576
* 577
* 578
* 579
* 580
* 581
* 582
* 583
* 584
* 585
* 586
* 587
* 588
* 589
* 590
* 591
* 592
* 593
* 594
* 595
* 596
* 597
* 598
* 599
* 600
* 601
* 602
* 603
* 604
* 605
* 606
* 607
* 608
* 609
* 610
* 611
* 612
* 613
* 614
* 615
* 616
* 617
* 618
* 619
* 620
* 621
* 622
* 623
* 624
* 625
* 626
* 627
* 628
* 629
* 630
* 631
* 632
* 633
* 634
* 635
* 636
* 637
* 638
* 639
* 640
* 641
* 642
* 643
* 644
* 645
* 646
* 647
* 648
* 649
* 650
* 651
* 652
* 653
* 654
* 655
* 656
* 657
* 658
* 659
* 660
* 661
* 662
* 663
* 664
* 665
* 666
* 667
* 668
* 669
* 670
* 671
* 672
* 673
* 674
* 675
* 676
* 677
* 678
* 679
* 680
* 681
* 682
* 683
* 684
* 685
* 686
* 687
* 688
* 689
* 690
* 691
* 692
* 693
* 694
* 695
* 696
* 697
* 698
* 699
* 700
* 701
* 702
* 703
* 704
* 705
* 706
* 707
* 708
* 709
* 710
* 711
* 712
* 713
* 714
* 715
* 716
* 717
* 718
* 719
* 720
* 721
* 722
* 723
* 724
* 725
* 726
* 727
* 728
* 729
* 730
* 731
* 732
* 733
* 734
* 735
* 736
* 737
* 738
* 739
* 740
* 741
* 742
* 743
* 744
* 745
* 746
* 747
* 748
* 749
* 750
* 751
* 752
* 753
* 754
* 755
* 756
* 757
* 758
* 759
* 760
* 761
* 762
* 763
* 764
* 765
* 766
* 767
* 768
* 769
* 770
* 771
* 772
* 773
* 774
* 775
* 776
* 777
* 778
* 779
* 780
* 781
* 782
* 783
* 784
* 785
* 786
* 787
* 788
* 789
* 790
* 791
* 792
* 793
* 794
* 795
* 796
* 797
* 798
* 799
* 800
* 801
* 802
* 803
* 804
* 805
* 806
* 807
* 808
* 809
* 810
* 811
* 812
* 813
* 814
* 815
* 816
* 817
* 818
* 819
```

```
* 33      endif UNIT NAME IS NOT IN DATA BASE          1820
* 34      endif INPUT IS NULL                         1821
* 35
* 36
* 37
* 38      PUPRT FOR START TIME                         1822
* 39      PUPRT FOR RISK AND TIME PARAMETERS ...C AND K 1823
* 40      CALCULATE BEST PATH                         1824
* 41      COMPUTE SOLUTION-VECTORS AND ROUTE-VECTOR    1825
* 42
* 43      SYSTEMATICALLY ERASE NODES, AND FORCE UNIT TO TRAVEL 1826
* 44      do FOR THE SECOND THROUGH NEXT-TO-LAST NODES OF THE BEST PATH 1827
* 45      ERASE NODE                                     1828
* 46      do FOR EACH NODE FROM FIRST NODE TO PREDECESSOR OF ERASED NODE 1829
* 47      LOOK IN SOLUTION-VECTOR OF NODE TO OBTAIN START TIME 1830
* 48
* 49
* 50      .WE'LL ONLY CALCULATE FROM THIS NODE TO THE DESTINATION 1831
* 51
* 52      CALCULATE BEST PATH OF THIS NODE TO THE DESTINATION 1832
* 53      IF PATH'S WORTH IS GREATER THAN ANY OF THESE IN THE ORIGIN 1833
* 54      PLACE THIS WORTH IN WORTH ARRAY                1834
* 55      REMEMBER INTERDICTED NODE                      1835
* 56      REMEMBER 'FORCED TO' NODE                     1836
* 57      endif PATH'S WORTH IS GREATER THAN ANY OF THOSE IN THE ARRAY 1837
* 58      enddo FOR EACH NODE FROM FIRST NODE TO PREDECESSOR OF ERASED NODE 1838
* 59      RESTORE ERASED NODE                         1839
* 60      enddo FOR THE SECOND THROUGH NEXT-TO-LAST NODE OF BEST PATH 1840
* 61
* 62      .CALCULATE CHOSEN PATHS. EACH PATH WILL BE THE COMBINATION OF TWO 1841
* 63      .PATHS, ONE FROM THE FIRST NODE TO THE INTERDICTED NODE, THEN ANOTHER 1842
* 64      .FROM THE INTERDICTED NODE TO THE DESTINATION.           1843
* 65
* 66      do FOR EACH ENTRY IN WORTH ARRAY                1844
* 67      ERASE REMEMBERED NODE                         1845
* 68      INITIALIZE *WORKING LIST* NODE-VECTOR ENTRIES 1846
* 69      CALCULATE BEST PATH FROM START NODE TO PREDECESSOR OF ERASED NODE 1847
* 70      COMPUTE SOLUTION-VECTORS AND ROUTE-VECTOR        1848
* 71
* 72
* 73
* 74
* 75
* 76
* 77
* 78
* 79
* 80
* 81
* 82
* 83
* 84
* 85
* 86
* 87
* 88
* 89
* 90
* 91
* 92
* 93
* 94
* 95
* 96
* 97
* 98
* 99
* 100
* 101
* 102
* 103
* 104
* 105
* 106
* 107
* 108
* 109
* 110
* 111
* 112
* 113
* 114
* 115
* 116
* 117
* 118
* 119
* 120
* 121
* 122
* 123
* 124
* 125
* 126
* 127
* 128
* 129
* 130
* 131
* 132
* 133
* 134
* 135
* 136
* 137
* 138
* 139
* 140
* 141
* 142
* 143
* 144
* 145
* 146
* 147
* 148
* 149
* 150
* 151
* 152
* 153
* 154
* 155
* 156
* 157
```

TRW, INC.
21 FEB 78

LOCATION AND MOVEMENT ANALYSIS SYSTEM
LAMAS PLOW

PAGE 62.002

```
* 71      INITIALIZE "WORKING LIST" NODE+VECTOR ENTRIES          * 1858
* 72      CALCULATE BEST PATH FROM PREDECESSOR TO DESTINATION NODE    * 1859
* 73      COMPUTE SOLUTION+VECTORS AND ROUTE+VECTOR                 * 1860
* 74      COMBINE SOLUTION+VECTORS AND ROUTE+VECTORS TO MAKE ONE ROUTE   * 1861
* 75      CREATE NEW UNIT ASSOCIATED WITH THIS ROUTE                  * 1862
* 76      PRINT MAP+NUMBER, NODE+NUMBER OF INTERDISCIPLINED NODE        * 1863
* 77      PRINT MAP+NUMBER, NODE+NUMBER OF "FORCED" NODE                * 1864
* 78      PRINT TIME LINDA                                         * 1865
* 79      PRINT TIME DELAY                                         * 1866
* 80      enddo FOR EACH ENTRY IN MURTH ARRAY                         * 1867
* 81      return TO CALLING PROGRAM                                * 1868
* 82      enddo UPON FUNCTION REQUEST                               * 1869
* 83
```

***** 100 MANY LINES IN SEGMENT

PRESENT RESULTS

REF	PAGE
*	*****
*	1 THESE ROUTINES ENABLE THE USER TO SEE THE RESULTS OF THE PATH ALGO-
*	2 .RITHMS AS PRINTED OUTPUT.
*	3 *
*	4 do UPON FUNCTION REQUEST
*	5 do FOREVER
*	6 DISPLAY FUNCTION MENU
*	7 WAIT FOR USER INPUT
*	8 do CASE OF
*	9 TABLE:
54 *	10 PRINT TABLE OF ROUTE NUMBERS WITH ASSOCIATED UNIT NAMES
65 *	11 HARDCOPY:
*	12 PRINT PATH STATISTICS
*	13 EXIT:
*	14 return TO CALLING PROGRAM
*	15 enddo CASE OF
*	16 enddo FOREVER
*	17 enddo UPON FUNCTION REQUEST
*	*****

TRI, INC.
LOCATION AND MOVEMENT ANALYSIS SYSTEM
LAMAS FLOW
21 FEB 78

PAGE 64

PRINT TABLE OF ROUTE NUMBERS WITH ASSOCIATED UNIT NAMES

REF PAGE

```
*****  
* 1 * IF THIS FUNCTION IS REQUESTED, A TABLE SHOWING ALL ROUTE NUMBERS, EACH  
* 2 * WITH THE UNIT NAME WITH WHICH IT IS ASSOCIATED, WILL BE PRINTED.  
* 3 *  
* 4 do UPON FUNCTION REQUEST  
* 5   PRINT TABLE OF ROUTE NUMBERS AND ASSOCIATED UNIT NAMES  
* 6   return TO CALLING PROGRAM  
* 7 enddo UPON FUNCTION REQUEST  
*****
```

AD-A069 029

TRW DEFENSE AND SPACE SYSTEMS GROUP REDONDO BEACH CALIF
LAMAS SYSTEM MANUAL. (U)

FEB 78

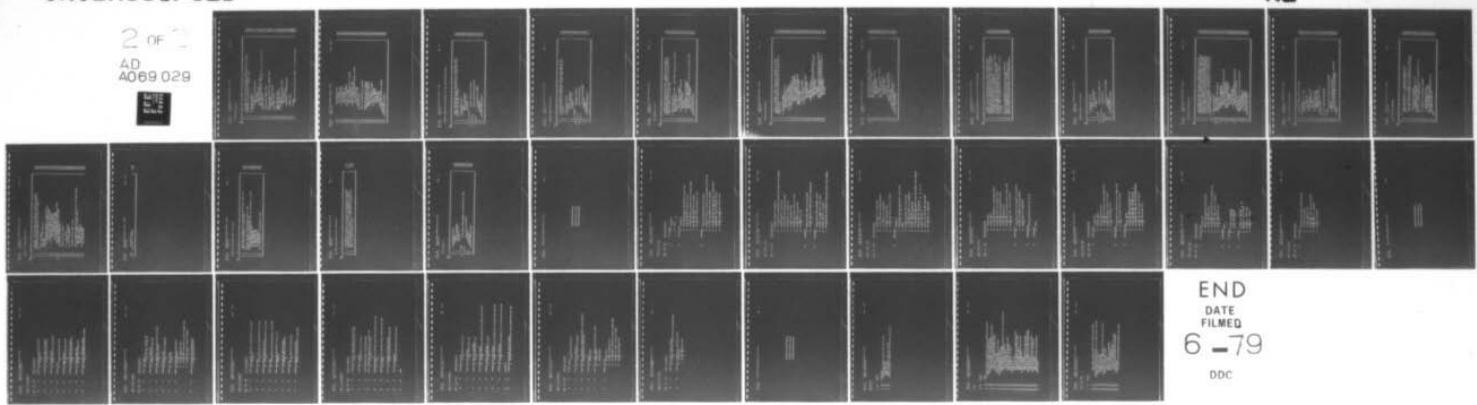
F/G 9/2

UNCLASSIFIED

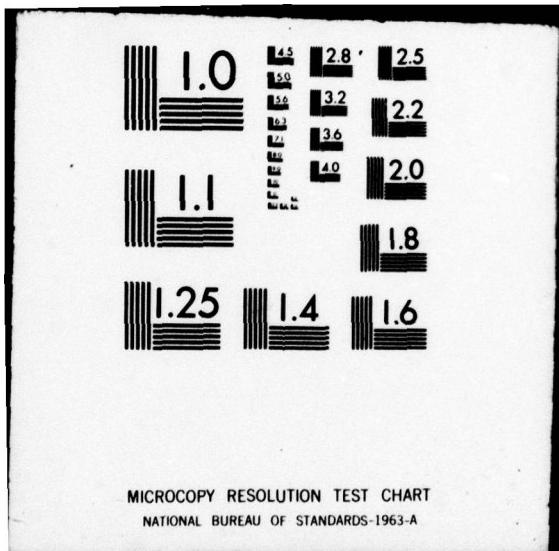
DAAG39-77-C-0112

NL

2 OF 2
AD
A069 029



END
DATE
FILED
6 -79
DDC



PPT11T PATH STATISTICS

REF

PAGE

```
* THIS ENABLE THE USER TO HAVE HARD-COPY EVIDENCE OF THE CALCULATIONS
* PERFORMED BY THIS SYSTEM. THIS INCLUDES TOTAL PATH STATISTICS, AS WELL
* AS THE ABILITY TO HAVE OUTPUT IN A SNAPSHOT MODE.
*
* 1
* 2
* 3
* 4
*
* 5 do UPON FUNCTION REQUEST
*   do FOR EACH UNIT
*     IF THERE IS NO ASSOCIATED ROUTE
*       PRINT MESSAGE
*       CYCLE FOR EACH UNIT
*       ENDIF THERE IS NO ASSOCIATED ROUTE
*       PRINT TOTAL PATH STATISTICS ...DISTANCE, TIME, STREET AND FLOOR, ETC.
*   enddo FOR EACH UNIT
*
* 6
* 7
* 8
* 9
* 10
* 11
* 12
* 13
* 14
* 15
* 16
* 17
* 18
* 19
* 20
* 21
* 22
* 23
* 24
* 25
* 26
* 27
* 28
* 29
* 30
* 31
* 32
*
* NOW SEE ABOUT SNAPSHOT MODE
*
* PROMPT FOR USER DESIRE TO HAVE SNAPSHOT OUTPUT
* IF INPUT IS NULL
*   RETURN TO CALLING PROGRAM
* ENDIF INPUT IS NULL
*
* PROMPT FOR START TIME
* PROMPT FOR TIME INCREMENT
* do FOR EACH UNIT
*   IF THERE IS NO ASSOCIATED ROUTE
*     PRINT MESSAGE
*     CYCLE FOR EACH UNIT
*   ENDIF THERE IS NO ASSOCIATED ROUTE
*
* SNAPSHOT MODE
*
* GO FOREVER
*   FIND UNIT POSITION FOR TIME OF INTEREST ...AS IN SNAPSHOT DISPLAY
*
* 1897
* 1898
* 1899
* 1900
* 1901
* 1902
* 1903
* 1904
* 1905
* 1906
* 1907
* 1908
* 1909
* 1910
* 1911
* 1912
* 1913
* 1914
* 1915
* 1916
* 1917
* 1918
* 1919
* 1920
* 1921
* 1922
* 1923
* 1924
* 1925
* 1926
* 1927
* 1928
```

* THIS ENABLE THE USER TO HAVE HARD-COPY EVIDENCE OF THE CALCULATIONS
* PERFORMED BY THIS SYSTEM. THIS INCLUDES TOTAL PATH STATISTICS, AS WELL
* AS THE ABILITY TO HAVE OUTPUT IN A SNAPSHOT MODE.

```
* 33      * UNIT HAS NOT STARTED          1929
* 34      * PRINT MESSAGE                1930
* 35      * cycle FOR EACH UNIT           1931
* 36      * elseif UNIT HAS FINISHED       1932
* 37          PRINT MESSAGE             1933
* 38          CYCLE FOR EACH UNIT        1934
* 39          endif UNIT HAS NOT STARTED   1935
* 40          if UNIT IS AT A NODE         1936
* 41              PRINT STATISTICS FOR UNIT AT THE NODE 1937
* 42          cycle FOR EACH UNIT         1938
* 43          else UNIT IS BETWEEN TWO NODES    1939
* 44              PRINT STATISTICS FOR UNIT BETWEEN THE NODES 1940
* 45          cycle FOR EACH UNIT         1941
* 46          endif UNIT IS AT A NODE       1942
* 47      enddo FOR EACH UNIT           1943
* 48
* 49      .SEE IF THE USER WISHES TO CONTINUE 1944
* 50
* 51      prompt FOR USER DESIRE TO CONTINUE 1945
* 52      if USER DOES NOT WANT TO CONTINUE 1946
* 53          prompt FOR NEW TIME INCREMENT 1947
* 54          if INPUT IS NULL             1948
* 55              return TO CALLING PROGRAM 1949
* 56          else INPUT IS NOT NULL        1950
* 57              SET TIME INCREMENT      1951
* 58              INCREMENT TIME        1952
* 59          cycle FOREVER            1953
* 60          endif INPUT IS NULL        1954
* 61          if USER DOES NOT WANT TO CONTINUE 1955
* 62      enddo FOREVER               1956
* 63      enddo LFIN FUNCTION REQUEST     1957
* 64
* 65
* 66
* 67
* 68
* 69
* 70
* 71
* 72
* 73
* 74
* 75
* 76
* 77
* 78
* 79
* 80
* 81
* 82
* 83
* 84
* 85
* 86
* 87
* 88
* 89
* 90
* 91
* 92
* 93
* 94
* 95
* 96
* 97
* 98
* 99
* 100
* 101
* 102
* 103
* 104
* 105
* 106
* 107
* 108
* 109
* 110
* 111
* 112
* 113
* 114
* 115
* 116
* 117
* 118
* 119
* 120
* 121
* 122
* 123
* 124
* 125
* 126
* 127
* 128
* 129
* 130
* 131
* 132
* 133
* 134
* 135
* 136
* 137
* 138
* 139
* 140
* 141
* 142
* 143
* 144
* 145
* 146
* 147
* 148
* 149
* 150
* 151
* 152
* 153
* 154
* 155
* 156
* 157
* 158
* 159
```

IMPLEMENT ALGORITHMS AND FUNCTIONS USING THE CROSS-COUNTRY DATA BASE

REF

PAGE

```
*****  
* 1 * THIS IS THE OTHER HALF OF THE LAMAS SYSTEM. IT IS MUCH MORE LIMITED  
* 2 * IN SCOPE THAN THE ROAD NETWORK SECTION, AND THIS IS MAINLY DUE TO THE  
* 3 * NATURE OF THE DATA BASE INVOLVED.  
* 4 *  
* 5 * do UPON FUNCTION REQUEST  
* 6 *   do FOREVER  
* 7 *     DISPLAY FUNCTION MENU  
* 8 *     WAIT FOR USER INPUT  
* 9 *     do CASE OF  
* 10 * INITIALIZE:  
* 11 *     PERFORM TERRAIN INITIALIZATION  
* 12 * PATHS:  
* 13 *     PERFORM TERRAIN PATH CALCULATIONS  
* 14 * EXIT:  
* 15 *     return TO CALLING PROGRAM  
* 16 *     enddo CASE OF  
* 17 *   enddo FOREVER  
* 18 * enddo UPON FUNCTION REQUEST  
*  
*****
```

1961

1962

1963

1964

1965

1966

1967

1968

1969

1970

1971

1972

1973

1974

1975

1976

1977

1978

TRI: INC.
21 FEB 78

LOCATION AND MOVEMENT ANALYSIS SYSTEM
LARAS FLIN

PAGE 57

PERFORM TERRAIN INITIALIZATION

REF PAGE

```
** IN THE CROSS-COUNTRY AND CONCEALMENT SECTION, THERE ARE ONLY TWO
** THINGS WHICH NEED TO BE INITIALIZED, THE MAP-NUMBER, AND THE UNITS
** DATA BASE.

** 4 do UPDN FUNCTION REQUEST
** 5   do FOREVER
** 6     DISPLAY FUNCTION MENU
** 7     WAIT FOR USER INPUT
** 8     do CASE OF
** 9       MAP-NUMBER:
** 10      MAP-NUMBER TO BE USED
** 11      12 UNITS:
** 12      ESTABLISH UNIT-VECTORS
** 13
** 14 EXIT:
** 15   return TO CALLING PROGRAM
** 16   enddo CASE OF
** 17   enddo FOREVER
** 18 enddo UPDN FUNCTION REQUEST
```

MAP NUMBER TO BE USED

REF	PAGE
1	1999
2	2000
3	2001
4	2002
5	2003
6	2004
7	2005
8	2006
9	2007
10	2008
11	2009
12	2010
13	2011
14	2012
15	2013
16	2014
17	2015
18	2016
19	E017

FOR THE CROSS-COUNTRY AND CONCEALMENT ALGORITHMS, ONLY ONE MAP MAY
BE CONSIDERED AT ONE TIME. THIS IS SO BECAUSE OF THE LARGE NUMBER OF
NODES WHICH ONE MAP REPRESENTS (9186), AND THE TIME INVOLVED FOR A
PATH CALCULATION USING THAT MANY NODES.

DO UPON FUNCTION REQUEST
READ "2CMDIR.DAT" INTO MEMORY ... CROSS-COUNTRY AND CONCEALMENT DIRECTORY
PROMPT FOR MAP+NUMBER
IF INPUT IS NULL
RETURN TO CALLING PROGRAM
ELSEIF INPUT NUMBER IS NOT IN DIRECTORY
PRINT ERROR MESSAGE
TRY AGAIN
ELSE INPUT NUMBER IS IN THE DIRECTORY
READ THE MAP'S NODES INTO MEMORY
ESTABLISH THE SCREEN EXTREMES ... VALUES ARE IN THE DIRECTORY ENTRY
RETURN TO CALLING PROGRAM
ENDIF INPUT IS NULL
ENDDO UPON FUNCTION REQUEST

ESTABLISH UNIT-VECTORS

REF PAGE

```
* JUST AS IN THE ROAD NETWORK SECTION, THIS DATA BASE MUST BE CON-
* STRUCTED BY THE USER. THE ONLY DIFFERENCE BETWEEN THE TWO IS THAT
* HERE, THE START AND DESTINATION ARE DESCRIBED AS UTM LATITUDE AND LON-
* GITUDE PAIRS.
*
* do UPON FUNCTION REQUEST
*
*    if NO MAP HAS BEEN INITIALIZED
*        PRINT ERROR MESSAGE
*        return TO CALLING PROGRAM
*
*    else A MAP HAS BEEN INITIALIZED
*        do UNTIL UNIT+COUNTER EQUALS 60
*            if USER DESIRES RE-INITIALIZATION
*                CLEAR ALL EXISTING UNIT-VECTORS
*                RESET UNIT+COUNTER TO ZERO
*            endif USER DESIRES RE-INITIALIZATION
*            PROMPT FOR ENTRY OF UNIT NAME
*            if INPUT IS NULL
*                return TO CALLING PROGRAM
*            else INPUT IS NOT NULL
*                if THIS UNIT NAME ALREADY EXISTS
*                    PRINT ERROR MESSAGE
*                    TRY AGAIN
*
*            else UNIT NAME IS NEW
*                PROMPT FOR UTM COORDINATES OF STARTING NODE
*                if COORDINATES DO NOT LIE WITHIN THE INITIALIZED MAP
*                    PRINT ERROR MESSAGE
*                    TRY AGAIN
*
*            endif COORDINATES DO NOT LIE WITHIN THE INITIALIZED MAP
*                PROMPT FOR UTM COORDINATE OF DESTINATION NODE
*                if COORDINATES DO NOT LIE WITHIN THE INITIALIZED MAP
*                    PRINT ERROR MESSAGE
*                    TRY AGAIN
*
*    2019
*    2020
*    2021
*    2022
*    2023
*    2024
*    2025
*    2026
*    2027
*    2028
*    2029
*    2030
*    2031
*    2032
*    2033
*    2034
*    2035
*    2036
*    2037
*    2038
*    2039
*    2040
*    2041
*    2042
*    2043
*    2044
*    2045
*    2046
*    2047
*    2048
*    2049
*    2050
```

TRE, INC.
21 FEB 73

LOCATION AND MOVEMENT ANALYSIS SYSTEM
LEMAS CLIN

PAGE 69.00:

```
* 33
70 * 34      endif COORDINATES DO NOT LIE WITHIN THE INITIALIZED MAP
    * 35      CHANGE ENTERED COORDINATES TO AN INDEX INTO THE DATA BASE
    * 36      MAKE ASSIGNMENTS TO UNIT VECTOR
    * 37      PROMPT FOR PRIORITY
    * 38      if PRIORITY NOT BETWEEN 0 AND 127
    * 39          PRINT ERROR MESSAGE
    * 40      TRY AGAIN
    * 41      endif PRIORITY NOT BETWEEN 0 AND 127
    * 42          PRINT FOR TYPE CODE
    * 43          if TYPE CODE NOT ALLOWABLE E
    * 44              PRINT ERROR MESSAGE
    * 45          TRY AGAIN
    * 46          endif TYPE CODE NOT ALLOWABLE
    * 47          endif THIS UNIT NAME ALREADY EXISTS
    * 48          endif INPUT IS NULL
    * 49      enddo UNTIL UNIT COUNTER EQUALS SIXTY
    * 50      endif NO MAP HAS BEEN INITIALIZED
    * 51      enddo UNTIL FUNCTION REQUEST
    *
```

CHANGE ENTERED COORDINATES TO AN INDEX INTO THE DATA BASE

REF PAGE

- 1 * IN ORDER TO KEEP THE MEMORY USAGE AT A MINIMUM, AN INDEXING SCHEME
2 * HAS BEEN DEVISED. THE DATA BASE HAS BEEN PACKED TOGETHER SO THAT
3 * ONE WORD CONTAINS FOUR DATA VALUES, OR EVERY DATA VALUE IS REPRESENTED
4 * BY FOUR BITS. ONE FOUR-BIT AREA IS CALLED A NIBBLE (HALF OF A BYTE).
5 * THUS, THERE ARE 8194 NIBBLES IN THE DATA BASE.
6 * THE POINT IS THEN TO MAP THE UTM COORDINATE FAIR TO ONE SQUARE ON THE
7 * .93 X .93 GRID, AND THEN TRANSLATE THAT VALUE TO A NIBBLE VALUE.
8 * THE POSSIBLE RANGE OF THE UTM COORDINATE PAIRS IS 1 TO 505
9 * (.22 * 23, = SQUARE KILOMETERS ON A MAP), SO WE'RE MAPPING 1 TO 506 TO
10 * 1 TO 8184.
11 * CALCULATE THE VALUE BETWEEN 1 AND 506 FOR THE GIVEN COORDINATES
12 * CALCULATE THE PERCENTAGE OF THIS VALUE OF THE WHILE
13 * MULTIPLY THE PERCENTAGE BY 8183 AND ADD 1 TO OBTAIN THE NIBBLE
14 * ASSIGN THE VALUE TO EITHER STARTING LOCATION OR DESTINATION LOCATION
15 *

2070 **
2071 **
2072 **
2073 **
2074 **
2075 **
2076 **
2077 **
2078 **
2079 **
2080 **
2081 **
2082 **
2083 **
2084 **

TRN, INC.
LOCATION AND MOVEMENT ANALYSIS SYSTEM
21 FEB 78
LAMAS FILM

PAGE 71

PERFORM TERRAIN PATH CALCULATIONS

REF	PAGE	CODE	COMMENT
		** 1	do UPON FUNCTION REQUEST
		** 2	do FOREVER
		** 3	DISPLAY FUNCTION MENU
		** 4	WAIT FOR USER INPUT
		** 5	do CASE OF
		** 6	PATH:
		** 7	CALCULATE CROSS-COUNTRY PATH
		** 8	SOLUTION:
		78 ** 9	DISPLAY RESULTS
		** 10	EXIT:
		** 11	return TO THE CALLING PROGRAM
		** 12	enddo CASE OF
		** 13	enddo FOREVER
		** 14	enddo UPON FUNCTION REQUEST

REF
PAGE

TRI, INC.
21 FEB 78

LOCATION AND MOVEMENT ANALYSIS SYSTEM
LANES FILE

CALCULATE CROSS-COUNTRY PATH

REF
PAGE

```
1 * THIS ROUTINE CALCULATES THE PATH OF A USER-ENTERED UNIT USING THE
2 * CROSS-COUNTRY DATA BASE. THE MAJOR DIFFERENCE BETWEEN THIS ROUTINE AND
3 * THE ROUTINE FOR THE ROAD NETWORK IS THAT THERE IS NOW AN ARRAY CALLED
4 * THE WORKING LIST. PREVIOUSLY, THE WORKING LIST VARIABLES WERE STORED
5 * WITHIN THE NODE+VECTOR, BUT NOW, WITH 8184 NODES TO BE CONSIDERED, IT
6 * IS TOTALLY IMPRactical TO TRY TO RETAIN SPACE FOR EACH NODE'S WORKING
7 * LIST ENTIRELY. Thus, A WORKING LIST ARRAY EXISTS. IT IS TREATED AS A
8 * LIST OF VECTORS, BEING USED IN ORDER (FROM INDEX 1 TO N) AS THEY ARE
9 * NEEDED BY THE ALGORITHM.
10 *
11 go JPN FUNCTION REQUEST
12 PURGE ALL ROUTES
13 OBTAIN UNIT NAMES TO BE CONSIDERED
14 do FOR EACH UNIT
15   OBTAIN THE PATH PARAMETERS
16   if THERE IS AN ERROR RETURN
17   GO BACK TO OBTAIN UNIT NAMES AND TRY AGAIN
18   endif THERE IS AN ERROR RETURN
19   CALCULATE THE CROSS-COUNTRY PATH
20   if THE PATH COULD NOT COMPLETE
21     PRINT MESSAGE
22     cycle FOR EACH UNIT
23   else THE PATH DID COMPLETE
24     CONSTRUCT SOLUTION+VECTORS AND ROUTE+VECTOR
25     endif THE PATH COULD NOT COMPLETE
26     senddo FOR EACH UNIT
27       PROMPT FOR USER WISH: FOR SOLUTION+VECTOR PRINTOUT
28       if USER WANTS OUTPUT
29         PRINT SOLUTION+VECTORS
30       endif? USER WANTS OUTPUT
31       return TO CALLING PROGRAM
32     enddo UPON FUNCTION REQUEST
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
```

TRI, INC.
21 FEB 78
LOCATION AND MOVEMENT ANALYSIS SYSTEM
LINES FLOW

PAGE 73

RETAIN UNIT NAMES TO BE CONSIDERED

PAGE

```
* 1 * THE USER MAY ENTER AT MOST SIXTY NAMES.
* 2 *
* 3 do SIXTY TIMES
* 4   PROMPT ENTRY OF UNIT NAME
* 5   if NULL INPUT
* 6     Undo SIXTY TIMES
* 7     endif NULL INPUT
* 8     if INPUT NAME DOES NOT EXIST
* 9       PRINT ERROR MESSAGE
*10      TRY AGAIN
*11    else UNIT NAME DOES EXIST
*12      INCREMENT UNIT COUNTER
*13      REMEMBER THE UNIT NAME
*14    endif INPUT NAME DOES NOT EXIST
*15  enddo SIXTY TIMES
*16 *
*17 .ALL UNIT NAME HAVE BEEN ENTERED.
*18
*19 do DNEC
*20   if LESS THAN TWO UNIT NAMES HAVE BEEN ENTERED
*21     Undo DNEC
*22   else MORE THAN ONE NAME HAS BEEN ENTERED
*23     SORT UNIT NAMES ACCORDING TO THEIR PRIORITY ... BUBBLE SORT
*24   endif LESS THAN TWO UNIT NAMES HAVE BEEN ENTERED
*25
*26 end do DNEC
```

*

OBTAI N THE PATH PARAMETERS

REF PAGE

```
* 1 do * 2 PROMPT FOR MOVEMENT TYPE ... BACKWARDS OR FORWARDS IN TIME  
* 3 PROMPT FOR START TIME ... DEPENDENT UPON MOVEMENT TYPE  
* 4 *  
* 5 .REINITIALIZATION OF THE WORKING LIST IS NEEDED ONLY IF EITHER THE  
* 6 .UNIT TYPE, START TIME, MOVEMENT TYPE, OR START LOCATION HAS CHANGED.  
* 7  
* 8 if REINITIALIZATION IS NEEDED  
* 9 SET WORKING LIST ARRAY ENTRIES TO ZERO  
* 10 endif REINITIALIZATION IS NEEDED  
* 11 ESTABLISH THE BEGINNING NODE (<=K), AND THE DESTINATION NODE (<=I)>  
* 12 SET J TO K ... J WILL TRACE THE PATH  
* 13  
* 14 .SET CUMULATIVE TIME AND WORTH FOR NODE J.  
* 15  
* 16 if J HAS NO WORKING LIST ENTRY  
* 17 CREATE NEW WORKING LIST ENTRY  
* 18 endif J HAS NO WORKING LIST ENTRY  
* 19 SET CUMULATIVE TIME TO ENTERED BEGINNING TIME  
* 20 SET CUMULATIVE WORTH TO 0  
* 21 enddo  
* 22  
* 23  
* 24  
* 25  
* 26  
* 27  
* 28  
* 29  
* 30  
* 31  
* 32  
* 33  
* 34  
* 35  
* 36  
* 37  
* 38  
* 39  
* 40  
* 41  
* 42  
* 43  
* 44  
* 45  
* 46  
* 47  
* 48  
* 49  
* 50  
* 51  
* 52  
* 53  
* 54  
* 55  
* 56  
* 57  
* 58  
* 59  
* 60  
* 61  
* 62  
* 63  
* 64  
* 65  
* 66  
* 67  
* 68  
* 69  
* 70  
* 71  
* 72  
* 73  
* 74  
* 75  
* 76  
* 77  
* 78  
* 79  
* 80  
* 81  
* 82  
* 83  
* 84  
* 85  
* 86  
* 87  
* 88  
* 89  
* 90  
* 91  
* 92  
* 93  
* 94  
* 95  
* 96  
* 97  
* 98  
* 99  
* 100  
* 101  
* 102  
* 103  
* 104  
* 105  
* 106  
* 107  
* 108  
* 109  
* 110  
* 111  
* 112  
* 113  
* 114  
* 115  
* 116  
* 117  
* 118  
* 119  
* 120  
* 121  
* 122  
* 123  
* 124  
* 125  
* 126  
* 127  
* 128  
* 129  
* 130  
* 131  
* 132  
* 133  
* 134  
* 135  
* 136  
* 137  
* 138  
* 139  
* 140  
* 141  
* 142  
* 143  
* 144  
* 145  
* 146  
* 147  
* 148  
* 149  
* 150  
* 151  
* 152  
* 153  
* 154  
* 155  
* 156  
* 157  
* 158  
* 159  
* 160  
* 161  
* 162  
* 163  
* 164  
* 165  
* 166  
* 167  
* 168  
* 169  
* 170  
* 171  
* 172  
* 173  
* 174  
* 175  
* 176  
* 177  
* 178  
* 179  
* 180
```

CALCULATE THE CROSS-COUNTRY PATH

```
* 1 * GIVEN NODE J, THE LAST NODE TO BE LABELED, THIS ROUTINE CALCULATES
* 2 * ITS ADJACENT NODES TIME VALUES. THEN THE NEXT NODE TO BE LABELED IS
* 3 * SELECTED AND THE PROCESS CONTINUES UNTIL NODE I IS LABELED.
* 4 *
* 5 do UNTIL I IS REACHED
* 6   FIND NODE I IN THE WORKING LIST
* 7   LABEL NODE J
* 8   do FOR EACH ADJACENT NODE OF J . . . EIGHT IN ALL
* 9     IF THE ADJACENT NODE HAS NO WORKING LIST ENTRY
* 10       CREATE AN ENTRY FOR THE ADJACENT NODE
* 11     else
* 12       IF THE ADJACENT NODE HAS A WORKING LIST ENTRY
* 13         IF THE ADJACENT NODE IS LABELED
* 14           CYCLE FOR EACH ADJACENT NODE
* 15         endif THE ADJACENT NODE IS LABELED
* 16       endif THE ADJACENT NODE HAS NO WORKING LIST ENTRY
* 17       CALCULATE THE ADJACENT NODE'S TIME VALUES
* 18       enddo FOR EACH ADJACENT NODE
* 19     .LOOK FOR I BEING LABELED
* 20   if I IS NOT LABELED
* 21     Undo UNTIL I IS REACHED
* 22     endif I IS NOW LABELED
* 23   .CALCULATE THE NEXT NODE TO LABEL
* 24   .
* 25   .
* 26   .
* 27   .
* 28   .
* 29   .
* 30   .
* 31   .
* 32   *
```

FIND THE UNLABELED WORKING LIST ENTRY WHICH HAS THE LEAST WORTH MEASURE
if NONE EXISTS .. THE ROUTE COULDN'T FINISH
PRINT ERROR MESSAGE
Undo UNTIL I IS REACHED
else AN ENTRY WAS FOUND
SET J TO THIS ENTRY

TRW, INC.
21 FEB 78

LOCATION AND MOVEMENT ANALYSIS SYSTEM
LAMAS FLOW

PAGE 75.001

```
* 33      endif NONE EXISTS
* 34      enddo UNTIL I IS REACHED
*
```

2214
*
*
*

CALCULATE THE ADJACENT NODE'S TIME VALUES

REF PAGE

```
*****  
* 1 AGAIN THIS IS SIMILAR TO THE TIME ROUTINE FOR THE ROAD NETWORK, AND  
* 2 AGAIN THE MAIN DIFFERENCE IS THE DATA BASE.  
* 3  
* 4 CALCULATE THE TIME OF TRAVEL BETWEEN J AND THIS NODE  
* 5 CALCULATE CUMULATIVE TIME FOR THIS LINK  
* 6 IF THE CUMULATIVE TIME CALCULATED IS LESS THAN THE WORKING LIST ENTRY  
* 7 SET NEW CUMULATIVE TIME  
* 8 SET NEW TIME MEASURE  
* 9 SET PREDECESSOR NODE  
* 10 SET CUMULATIVE WORTH  
* 11 SET NORTH MEASURE  
* 12 endif THE CUMULATIVE TIME CALCULATED IS LESS  
*****
```

```
*****  
* 17 2217  
* 18 2218  
* 19 2219  
* 20 2220  
* 21 2221  
* 22 2222  
* 23 2223  
* 24 2224  
* 25 2225  
* 26 2226  
* 27 2227  
* 28 2228  
*****
```

CONSTRUCT SOLUTION-VECTORS AND RROUTE-VECTOR

PAGE 77

REF PAGE

* 1 * THIS ROUTINE IS IDENTICAL TO THE ROAD NETWORK ROUTINE WHICH PERFORMS
* 2 * THE SAME TASK EXCEPT THAT THE WORKING LIST IS REFERENCED IN ORDER TO
* 3 * OBTAIN CERTAIN VALUES, SUCH AS 1) TOTAL WORTH MEASURE, 2) STARTING
* 4 * TIME, 3) TIME MEASURE (LINK TIME), 4) PREDECESSOR'S CUMULATIVE TIME,
* 5 * AND 5) THIS NODE'S CUMULATIVE TIME

* 2230

* 2231

* 2232

* 2233

* 2234

TRW, INC. LOCATION AND MOVEMENT ANALYSIS SYSTEM
21 FEB 78 LANS FLIN

DISPLAY RESULTS

REF PAGE

```
* 1 do UPON FUNCTION REQUEST          2236
* 2 do FOREVER                         2237
* 3 DISPLAY FUNCTION MENU             2238
* 4 WAIT FOR USR INPUT                2239
* 5 do CASE IF                         2240
* 6 TABLE:                            2241
* 7 PRINT TABLE OF ROUTE NUMBERS WITH ASSOCIATED UNIT NAMES 2242
* 8 EXIT:                             2243
* 9 return TO CALLING PROGRAM        2244
* 10 enddo CASE IF                   2245
* 11 enddo FOREVER                   2246
* 12 enddo UPDN FUNCTION REQUEST     2247
```

PAGE 78

TRW INC.
21 FEB 78

LOCATION AND MOVEMENT ANALYSIS SYSTEM

PAGE 79

* INDEX TO DATA ITEMS *

TEA, INC.
21 FEB 78 LOCATION AND MOVEMENT ANALYSIS SYSTEM
INDEX TO DATA ITEMS

PAGE 79.00!

INDEX TO DATA ITEMS

PAGE LINE TYPE

NAME AND REFERENCES

DI MAIN PROGRAM

26 PREPARATION

1

35 LAMAS

1 15

DI MAP+DIRECTORY

26 PREPARATION

2

27 CREATE NODE+VECTOR AND MAP+DIRECTORY FILES ON DISK

21 43 45

32 CREATE MAP+DIRECTORY ENTRY FOR THIS MAP

8 9

39 ESTABLISH DIRECTORY IN MEMORY AND READ IN NODE+VECTORS

12 16 17 18

40 READ THIS MAP'S NODES INTO MAIN MEMORY

18 19 20

41 CHANGE ADJACENT NODE'S MAP+NUMBER TO INDEX

16

DI MAP+NUMBER

32 CREATE MAP+DIRECTORY ENTRY FOR THIS MAP

8

33 CREATE CROSS-COUNTRY, CONCEALMENT, AND DIRECTORY FILES ON DISK

50

33 INITIALIZE MAP+NUMBERS AND SCREEN EXTREMES

5 6 7

39 ESTABLISH DIRECTORY IN MEMORY AND READ IN NODE+VECTORS

15 16 21 22 24 26 27

41 CHANGE ADJACENT NODE'S MAP+NUMBER TO INDEX

16

43 ESTABLISH A FORWARD EDGE OF BATTLE AREA

5 9 14

INDEX TO DATA ITEMS

PAGE	LINE	TYPE	NAME AND REFERENCES
	45		PERFORM INTERDUCTIVE OPERATIONS
	47		PRINT MAP+NUMBER, NODE+NUMBER OF NODE+VECTOR NEAREST GIVEN COORDINATE
	48		CHANGE CONTENTS OF A NODE+VECTOR
	50		DELETE A NODE FROM THE NODE+VECTOR ARRAY
	51		PRINT CONTENTS OF A NODE+VECTOR
	60		CALCULATE SECOND BEST PATH
	62		CALCULATE BEST NODE AT WHICH TO INTERDUCE
	67		PERFORM TERRAIN INITIALIZATION
	68		MAP+NUMBER TO BE USED
	69		
E1			MAP+NUMBERS
	37		PREPARE FOR ROAD NETWORK PATH CALCULATIONS
	39		ESTABLISH DIRECTORY IN MEMORY AND READ IN NODE+VECTORS
D1			NODE+NUMBER
	41		CHANGE ADJACENT NODE'S MAP+NUMBER TO INDEX
	43		ESTABLISH A FORWARD EDGE OF BATTLE AREA
	46		PERFORM INTERDUCTIVE OPERATIONS
	47		PRINT MAP+NUMBER, NODE+NUMBER OF NODE+VECTOR NEAREST GIVEN COORDINATE
	48		

INDEX TO DATA ITEMS

PAGE	LINE	TYPE	NAME AND REFERENCES
	43	CHANGE CONTENTS OF A NODE+VECTOR	7 12
	50	DELETE A NODE FROM THE NODE+VECTOR ARRAY	
	51	PRINT CONTENTS OF A NODE+VECTOR	9
	60	CALCULATE SECOND BEST PATH	2
	62	CALCULATE BEST NODE AT WHICH TO INTERDICT	62
	62	CALCULATE BEST NODE AT WHICH TO INTERDICT	76 77
DI	26	NODE+VECTOR PREPARATION	
	27	CREATE NODE+VECTOR AND MAP+DIRECTORY FILES ON DISK	2
	34	35 36 38 44	
	35	LAMAS	7
	43	ESTABLISH A FORWARD EDGE OF BATTLE AREA	
	44	FIND NODE+VECTOR NEAREST TO UTM COORDINATES	14 15 9 12 15
	46	PERFORM INTERDICTIVE OPERATIONS	11 13 15 17
	47	PRINT MAP+NUMBER, NODE+NUMBER OF NODE+VECTOR NEAREST GIVEN COORDINATE	7 11 13 17
	48	CHANGE CONTENTS OF A NODE+VECTOR	8 11 13 16 17 17 38 39 40
	49	CHANGE CHARACTERISTIC	7
	50	DELETE A NODE FROM THE NODE+VECTOR ARRAY	9
	51	PRINT CONTENTS OF A NODE+VECTOR	7

INDEX TO DATA ITEMS

PAGE	LINE	TYPE	NAME AND REFERENCES
	53		CALCULATE PATH
	54		PURGE ALL ROUTES
	54		13 15
	55		REINITIALIZE "WORKING LIST" NODE+VECTOR ENTRIES
	55		4 14
	56		CALCULATE AND ASSIGN TIME AND WORTH VALUES
	56		33 35
	59		COMPUTE SOLUTION+VECTORS AND ROUTE+VECTOR
	59		46 58
	60		CALCULATE SECOND BEST PATH
	60		42 57
	61		FIND THE ORDER OF PRIORITIES FOR BEST NETWORK TRAVEL
	61		49
	62		CALCULATE BEST NODE # WHICH TO INTERDICT
	62		68 71
DI		NODE+VECTORS	
DI	27		CREATE NODE+VECTOR AND MAP+DIRECTORY FILES ON DISK
DI	27		20
DI	33		INITIALIZE MAP+NUMBERS AND SCREEN EXTREMES
DI	48		2 CHANGE CONTENTS OF A NODE+VECTOR
DI		NUMBER+IF+MAPS	
DI	32		CREATE MAP+DIRECTORY ENTRY FOR THIS MAP
DI	32		9
DI		ROAD+NETWORK	
DI	35		LAMAS
DI	35		6

TRI, INC.
21 FEB 78

LOCATION AND MOVEMENT ANALYSIS SYSTEM
INDEX TO DATA ITEMS

INDEX TO DATA ITEMS

PAGE LINE TYPE

None and References

NI ROUTE+VECTOR

53 CALCULATE PATH
15

54 PURGE ALL ROUTES

6 8

60 CALCULATE SECOND BEST PATH

35 59

61 FIND THE ORDER OF PRIORITIES FOR BEST NETWORK TRAVEL

61 FIND 53 69

62 CALCULATE BEST NODE AT WHICH TO INTERDICT

41 20 73

72 CALCULATE CROSS-COUNTRY PATH

24

DI ROUTE+VECTORS

51 FIND THE ORDER OF PRIORITIES FOR BEST NETWORK TRAVEL
65

62 CALCULATE BEST NODE AT WHICH TO INTERDICT

74

DI SOLUTION+VECTOR

54 PURGE ALL ROUTES

9 11 14

56 CALCULATE AND ASSIGN TIME AND NORTH VALUES

40 42 44 46 49 67

59 COMPUTE SOLUTION+VECTORS AND ROUTE+VECTOR

24 25 46 47 48 53 54 58

61 FIND THE ORDER OF PRIORITIES FOR BEST NETWORK TRAVEL

49

62 CALCULATE BEST NODE AT WHICH TO INTERDICT

48

72 CALCULATE CROSS-COUNTRY PATH

27

INDEX TO DATA ITEMS

PAGE	LINE	TYPE	NAME AND REFERENCES
DI	SOLUTION+VECTORS		

53	CALCULATE PATH	15	
59	COMPUTE SOLUTION+VECTORS AND ROUTE+VECTOR	33 43 63	
60	CALCULATE SECOND BEST PATH	35 59	
61	FIND THE ORDER OF PRIORITIES FOR BEST NETWORK TRAVEL	53 66 69	
62	CALCULATE BEST NODE AT WHICH TO INTERDICT	41 70 73 74	
72	CALCULATE CROSS-COUNTRY PATH	24 29	
DI	TERRAIN+MODEL		
35	LAMAS	8	
DI	UNIT+COUNTER		
42	INITIALIZE UNIT+VECTORS	9 14 26 58	
69	ESTABLISH UNIT+VECTORS	11 14 48	
DI	UNIT+VECTOR		
42	INITIALIZE UNIT+VECTORS	27 33 34 40 46 52	
52	CHANGE PROPERTIES OF A UNIT	10	
60	CALCULATE SECOND BEST PATH	60 61	
69	ESTABLISH UNIT+VECTORS	35	

INDEX TO DATA ITEMS

PAGE LINE TYPE

NAME AND REFERENCES

DI	UNIT+VECTORS	37	PREPARE FOR ROAD NETWORK PATH CALCULATIONS
		9	
	42	INITIALIZE UNIT+VECTORS	
	60	8	CALCULATE SECOND BEST PATH
	16	19	CALCULATE BEST NODE AT WHICH TO INTERDICT
	62	22	PERFORM TERRAIN INITIALIZATION
	67	25	
	13	13	ESTABLISH UNIT+VECTORS
	69	13	

TRI, INC.
21 FEB 78

LOCATION AND MOVEMENT ANALYSIS SYSTEM

PAGE 89

* INDEX TO FLOW SEGMENTS *

INDEX TO FLOW SEGMENTS

PARF	LINE	TYPE	NAME AND REFERENCES
56	FS		CALCULATE AND ASSIGN TIME AND WORTH VALUES 53 CALCULATE PATH 08
62	FS		CALCULATE BEST NODE AT WHICH TO INTERDICT 45 PATH DETERMINATION AND DISPLAY 32
72	FS		CALCULATE CROSS-COUNTRY PATH 71 PERFORM TERRAIN PATH CALCULATIONS 7
53	FS		CALCULATE PATH 45 PATH DETERMINATION AND DISPLAY 26
60	FS		CALCULATE SECOND BEST PATH 45 PATH DETERMINATION AND DISPLAY 28
76	FS		CALCULATE THE ADJACENT NODE'S TIME VALUES 75 CALCULATE THE CROSS-COUNTRY PATH 16
75	FS		CALCULATE THE CROSS-COUNTRY PATH 72 CALCULATE CROSS-COUNTRY PATH 19
41	FS		CHANGE ADJACENT NODE'S MAP+NUMBER TO INDEX 38 INITIALIZE MAP+NUMBERS AND SCREEN EXTREMES 6

INDEX TO FLOW SEGMENTS

PAGE	LINE	TYPE	NAME AND REFERENCES
49		FS	CHANGE CHARACTERISTIC 48 CHANGE CONTENTS OF A NODE+VECTOR 25 27 29 31 33 35
48		FS	CHANGE CONTENTS OF A NODE+VECTOR 46 PERFORM INTERDICTIVE OPERATIONS 13
70		FS	CHANGE ENTERED COORDINATES TO AN INDEX INTO THE DATA BASE 69 ESTABLISH UNIT+VECTORS 34
52		FS	CHANGE PROPERTIES OF A UNIT 46 PERFORM INTERDICTIVE OPERATIONS 19
57		FS	COMPUTE NEXT NODE TO LABEL 53 CALCULATE PATH 12
59		FS	COMPUTE SOLUTION+VECTORS AND ROUTE+VECTOR 53 CALCULATE PATH 15 60 CALCULATE SECOND BEST PATH 35 59 61 FIND THE ORDER OF PRIORITIES FOR BEST NETWORK TRAVEL 69 62 CALCULATE BEST NODE AT WHICH TO INTERJECT 41 70 73
77		FS	CONSTRUCT SOLUTION+VECTORS AND ROUTE+VECTOR 72 CALCULATE CROSS-COUNTRY PATH 24

INDEX TO FLOW SEGMENTS

PAGE	LINE	TYPE	NAME AND REFERENCES
33	FS		CREATE CROSS-COUNTRY, CONCEALMENT, AND DIRECTORY FILES ON DISK 26 PREPARATION 3
36	FS		CREATE LINK CARD 27 CREATE NODE+VECTOR AND MAP+DIRECTORY FILES ON DISK 10
32	FS		CREATE MAP+DIRECTORY ENTRY FOR THIS MAP 27 CREATE NODE+VECTOR AND MAP+DIRECTORY FILES ON DISK 43
28	FS		CREATE NODE CARD 27 CREATE NODE+VECTOR AND MAP+DIRECTORY FILES ON DISK ?
27	FS		CREATE NODE+VECTOR AND MAP+DIRECTORY FILES ON DISK 26 PREPARATION 2
50	FS		DELETE A NODE FROM THE NODE+VECTOR ARRAY 46 PERFORM INTERDUCTIVE OPERATIONS 15
78	FS		DISPLAY RESULTS 71 PERFORM TERRAIN PATH CALCULATIONS 9
43	FS		ESTABLISH A FORWARD EDGE OF BATTLE AREA 37 PREPARE FOR ROAD NETWORK PATH CALCULATIONS 11

INDEX TO FILM SEGMENTS

PAGE	LINE	TYPE	NAME AND REFERENCES
39		FS	ESTABLISH DIRECTORY IN MEMORY AND READ IN NODE+VECTORS 38 INITIALIZE MAP+NUMBERS AND SCREEN EXTREMES 2
60		FS	ESTABLISH UNIT+VECTORS 67 PERFORM TERRAIN INITIALIZATION
44		FS	FIND NODE+VECTOR NEAREST TO UTM COORDINATES
61		FS	FIND THE ORDER OF PRIORITIES FOR BEST NETWORK TRAVEL 45 PATH DETERMINATION AND DISPLAY 30
66		FS	IMPLEMENT ALGORITHMS AND FUNCTIONS USING THE CROSS-COUNTRY DATA BASE 35 LAMAS 9
36		FS	IMPLEMENT ALGORITHMS AND FUNCTIONS USING THE NODE+VECTOR DATA BASE 35 LAMAS 7
38		FS	INITIALIZE MAP+NUMBERS AND SCREEN EXTREMES 37 PREPARE FOR ROAD NETWORK PATH CALCULATIONS 7
42		FS	INITIALIZE UNIT+VECTORS 37 PREPARE FOR ROAD NETWORK PATH CALCULATIONS 9
35		FS	LAMAS

TRW, INC.
21 FEB 78

LOCATION AND MOVEMENT ANALYSIS SYSTEM
INDEX TO FLIN SEGMENTS

PAGE: 80.005

INDEX TO FLIN SEGMENTS

PAGE	LINE	TYPE	NAME AND REFERENCES
63	FS		MAP+NUMBER TO BE USED 67 PERFORM TERRAIN INITIALIZATION 11
74	FS		OBTAIN THE PATH PARAMETERS 72 CALCULATE CROSS-COUNTRY PATH 15
73	FS		OBTAIN UNIT NAMES TO BE CONSIDERED 72 CALCULATE CROSS-COUNTRY PATH 13
45	FS		PATH DETERMINATION AND DISPLAY 36 IMPLEMENT ALGORITHMS AND FUNCTIONS USING THE NODE-VECTOR DATA BASE 9
46	FS		PERFORM INTERDUCTIVE OPERATIONS 45 PATH DETERMINATION AND DISPLAY 24
67	FS		PERFORM TERRAIN INITIALIZATION 66 IMPLEMENT ALGORITHMS AND FUNCTIONS USING THE CROSS-COUNTRY DATA BASE 11
71	FS		PERFORM TERRAIN PATH CALCULATIONS 66 IMPLEMENT ALGORITHMS AND FUNCTIONS USING THE CROSS-COUNTRY DATA BASE 13
26	FS		PREPARATION
37	FS		PREPARE FOR ROAD NETWORK PATH CALCULATIONS 36 IMPLEMENT ALGORITHMS AND FUNCTIONS USING THE NODE-VECTOR DATA BASE 7

TRI, INC.
21 FEB 78

LOCATION AND MOVEMENT ANALYSIS SYSTEM
INDEX TO FLIN SEGMENTS

INDEX TO FLIN SEGMENTS

PAGE	LINE	TYPE	NAME AND REFERENCES
63	FS	PRESENT RESULTS	45 PATH DETERMINATION AND DISPLAY 34
51	FS	PRINT CONTENTS OF A NODE+VECTOR	46 PERFORM INTERDICTIVE OPERATIONS
47	FS	PRINT MAP+NUMBER, NODE+VECTOR NEAREST GIVEN COORDINATES	46 PERFORM INTERDICTIVE OPERATIONS 17
65	FS	PRINT PATH STATISTICS	63 PRESENT RESULTS 12
67	FS	PRINT TABLE OF ROUTE NUMBERS WITH ASSOCIATED UNIT NAMES	63 PRESENT RESULTS 10 78 DISPLAY RESULTS 7
54	FS	PURGE ALL ROUTES	53 CALCULATE PATH 26 50 CALCULATE SECOND BEST PATH 10 61 FIND THE ORDER OF PRIORITIES FOR BEST NETWORK TRAVEL 13 62 CALCULATE BEST NODE AT WHICH TO INTERDICT 16 72 CALCULATE CROSS-COUNTRY PATH 12

PAGE 80.006

TRW, INC. LOCATION AND MOVEMENT ANALYSIS SYSTEM
2: FEB 73 INDEX TO FLOW SEGMENTS

PAGE 80-507

INDEX TO FLOW SEGMENTS

PHASE	LINE	TYPE	NAME AND REFERENCES
40	FS	READ THIS MAP'S NODES INTO MAIN MEMORY 39 ESTABLISH DIRECTORY IN MEMORY AND READ IN NODE-VECTORS 20	
55	FS	REINITIALIZE "WORKING LIST" NODE-VECTOR ENTRIES 53 CALCULATE PATH 78 83 60 CALCULATE SECOND BEST PATH 42 57	

TRI, INC.
21 FEB 78

LOCATION AND MOVEMENT ANALYSIS SYSTEM

PAGE 81

*
* SEGMENT REFERENCE TREES *
*
*
*
*
*

TRW, INC.
21 FEB 78

LOCATION AND MOVEMENT ANALYSIS SYSTEM
SEGMENT REFERENCE TREES

PAGE 81.001

PREPARATION

LN	DEF	SEGMENT
1	26	PREPARATION
2	27	CREATE NODE-VECTOR AND MAP+DIRECTORY FILES ON DISK
3	28	CREATE NODE CARD
4	30	CREATE LINK CARD
5	32	CREATE MAP+DIRECTORY ENTRY FOR THIS MAP
6	33	CREATE CROSS-COUNTRY, CONCEALMENT, AND DIRECTORY FILES ON DISK

LAMAS

LN	NEF	SEGMENT	DESCRIPTION
1	35	LAMAS	IMPLEMENT ALGORITHMS AND FUNCTIONS USING THE NODE+VECTOR DATA BASE
2	36		PREPARE FOR ROAD NETWORK PATH CALCULATIONS
3	37		INITIALIZE MAP+NUMBERS AND SCREEN EXTREMES
4	38		ESTABLISH DIRECTORY IN MEMORY AND READ IN NODE+VECTORS
5	39		READ THIS MAP'S NODES INTO MAIN MEMORY
6	40		CHANGE ADJACENT NODE'S MAP+NUMBER TO INDEX
7	41		INITIALIZE UNITS+VECTORS
8	42		ESTABLISH A FORWARD EDGE OF BATTLE AREA
9	43		PATH DETERMINATION AND DISPLAY
10	45		PERFORM INTERDICTIVE OPERATIONS
11	46		PRINT MAP+NUMBER, NODE+NUMBER OF NODE+VECTOR NEAREST GIVEN COORDINATES
12	47		
13	48		
14	49		CHANGE CONTENTS OF A NODE+VECTOR
15	50		CHANGE CHARACTERISTIC
16	51		DELETE A NODE FROM THE NODE+VECTOR ARRAY
17	52		PRINT CONTENTS OF A NODE+VECTOR
18	53		CHANGE PROPERTIES OF A UNIT
19	54		CALCULATE PATH
20	55		PURGE ALL ROUTES
21	56		REINITIALIZE "WORKING LIST" NODE+VECTOR ENTRIES
22	57		CALCULATE AND ASSIGN TIME AND NORTH VALUES
23	58		COMPUTE NEXT NODE TO LABEL
24	59		COMPUTE SOLUTION+VECTORS AND ROUTE+VECTOR
25	60		CALCULATE SECOND BEST PATH
26	69		PURGE ALL ROUTES
27	55		COMPUTE SOLUTION+VECTORS AND ROUTE+VECTOR
28	61		REINITIALIZE "WORKING LIST" NODE+VECTOR ENTRIES
29	54		FIND THE ORDER OF PRIORITIES FOR BEST NETWORK TRAVEL
30	59		PURGE ALL ROUTES
31	62		COMPUTE SOLUTION+VECTORS AND ROUTE+VECTOR
32	54		CALCULATE BEST NODE AT WHICH TO INTERDICT
33	59		PURGE ALL ROUTES
34	63		COMPUTE SOLUTION+VECTORS AND ROUTE+VECTOR
			PRESENT RESULTS

TRI, INC.
21 FEB 78

LOCATION AND MOVEMENT ANALYSIS SYSTEM
SEGMENT REFERENCE TREES

LAMAS (CONTINUED)

PAGE: 81.003

LN	DEF	SEGMENT
35	64	PRINT TABLE OF ROUTE NUMBERS WITH ASSOCIATED UNIT NAMES
36	65	PRINT PATH STATISTICS
37	66	IMPLEMENT ALGORITHMS AND FUNCTIONS USING THE CROSS-COUNTRY DATA BASE
38	67	PERFORM TERRAIN INITIALIZATION
39	68	HAP+NUMBER TO BE USED
40	69	ESTABLISH UNIT+VECTORS
41	70	CHANGE ENTERED COORDINATES TO AN INDEX INTO THE DATA BASE
42	71	PERFORM TERRAIN PATH CALCULATIONS
43	72	CALCULATE CROSS-COUNTRY PATH
44	73	PURGE ALL ROUTES
45	74	DETAIN UNIT NAMES TO BE CONSIDERED
46	75	DETAIN THE PATH PARAMETERS
47	76	CALCULATE THE CROSS-COUNTRY PATH
48	77	CALCULATE THE ADJACENT NODE'S TIME VALUES
49	78	CONSTRUCT SOLUTION+VECTORS AND ROUTE+VECTOR
50	79	DISPLAY RESULTS
51	80	PRINT TABLE OF ROUTE NUMBERS WITH ASSOCIATED UNIT NAMES